Network Working Group
Internet-Draft
Intended status: Informational
Expires: January 10, 2008

W. Eddy, Ed.
Verizon
S. Bellovin
Columbia University
J. Touch
USC/ISI
R. Bonica
Juniper Networks
July 9, 2007

Problem Statement and Requirements for a TCP Authentication Option
draft-bellovin-tcpsec-01

Status of this Memo

Copyright Notice

Abstract

The TCP-MD5 option is commonly used to secure BGP sessions between routers, although it is known to have many serious deficiencies. This memo presents requirements for a TCP segment authentication mechanism that is intended to replace TCP-MD5. While TCP-MD5 was designed to protect TCP sessions whose payload is BGP, the applicability of the mechanism described herein is broader. This mechanism can be applied to any TCP connection, regardless of payload.

Table of Contents

1. Contributors

This document resulted from the discussions of several IETF participants, including significant input from a design team within the TCPM working group who included (alphabetically):

Mark Allman (mallman@icir.org)

Steve Bellovin (smb@cs.columbia.edu)

Ron Bonica (rbonica@juniper.net)

Wesley Eddy (weddy@grc.nasa.gov)

Andrew Lange (andrew.lange@alcatel.com)

Allison Mankin (mankin@psg.com)

Sandy Murphy (sandy@tislabs.com)

Joe Touch (touch@isi.edu)

Sriram Viswanathan (sriram_v@cisco.com)

Brian Weis (bew@cisco.com)

2. Introduction

Putting a security service into the transport layer has a long history. SP4 [SP4] [SP4P] provided that service for the Secure Data Network System (SDNS); OSI incorporated SP4 into its protocol suite as the Transport Layer Security Protocol (TLSP) [TLSP].

TCP/IP has not had a full-fledged equivalent, though the TCP-MD5 option [RFC2385] has served for some purposes. TCP-MD5 is now known to have several problems. In this memo, we analyze the need for a TCP-based security service in Section 3 and discuss the requirements that a solution should meet in Section 4,

Note that we have deliberately used the phrase "security service". The in-use TCP-MD5 provides authentication-only and no TCP-based confidentiality mechanism is deployed or yet defined by the IETF. This document focuses on the requirements for an authentication-only service to replace TCP-MD5. If a TCP-based confidentiality service is also warranted, it could share many of these requirements, but this is beyond the scope of the current work or expressed needs from the community.

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 [RFC2119].

3. Problem Statement

The TCP-MD5 mechanism described in [RFC2385], includes a Message Authentication Code (MAC) in each TCP header. The MAC value is computed by hashing over:

- the TCP pseudo-header
- the TCP header, excluding options, and assuming a checksum of zero
- the TCP segment data
- an independently-specified shared key or password

To successfully spoof segments to a connection using the scheme described above, an attacker would not only have to guess TCP sequence numbers, but would also have to obtain the key that was used to calculate the MAC. This key never appears in the connection stream.

[RFC3562] addresses key management considerations for TCP-MD5. Based upon the cryptographic strength of the MD5 hashing algorithm, RFC 3562 recommends that keys be changed at least every 90 days. Unfortunately, TCP-MD5 only permits keys to be changed during the lifetime of a TCP connection if the change is synchronized at both ends. This limitation has proven to be a deterrent to the effective deployment of TCP-MD5, and necessitates a heuristic for key change [RFC4808].

Also, TCP-MD5 is entirely dependent on the MD5 hash algorithm, for which there are now well-known collision-finding methods. In addition, the particular keyed-hash MAC construction used by TCP-MD5 has serious cryptographic weaknesses. An attacker who can find a collision in the underlying hash function can forge a MAC using a simple chosen-message attack. It is quite clear that the existing TCP-MD5 mechanism is inadequate [I-D.manral-rpsec-existing-crypto]. It is cryptographically unsound, requiring a process waiver to permit its continued use with BGP [RFC4278].

TCP-MD5 has also been accused of not meeting operator requirements, even though it was originally intended for operators to protect TCP-based routing protocol sessions with (e.g. BGP, and now also LDP). TCP-MD5 is said to have high CPU utilization. The impact of MD5 itself is known [RFC1810], but for specific TCP-MD5 implementations, hard data on the protocol's performance has not been made availble, nor have direct comparisons between TCP-MD5 and IPsec AH performance.

The key management and key change synchronization difficulties

mentioned above have also been raised as operator concerns. It has been admitted that many operators simply do not change keys on a regular or systematic basis, but it is not clear whether this is a symptom of TCP-MD5's lack of capabilities, or unrelated operational culture. Based on the importance to the Internet of security for routing protocol sessions, it is clear that TCP-MD5 should be improved upon, and it seems likely that an improved version could greatly increase the use of TCP-based authentication for routing protocols and thus the robustness of routing sessions against the known attacks targeting TCP connections [I-D.ietf-tcpm-tcp-antispoof] [I-D.ietf-tcpm-tcpsecure].

It is less clear why authentication is needed at all within TCP implementations. IPsec [RFC4301] can protect the entire TCP header and payload, and TLS [RFC4346] can protect the payload data within a TCP connection, when used by an application. However, these existing solutions have their own deficiencies [I-D.ietf-tcpm-tcp-antispoof].

The most serious problem with IPsec is that it is hard to protect an individual TCP connection with it [I-D.bellovin-useipsec], due to the lack of an API that an application can request IPsec protection for a specific connection via. IPsec operates at the IP layer (with only a sprinkling of transport layer concepts, such as port numbers used within traffic selectors), and has no notion of individual transport layer connections and their duration (only quintuples of IP addresses, protocol, and port numbers), so "latching" a particular TCP connection to an IPsec Security Association with a corresponding lifetime is difficult [I-D.ietf-btns-connection-latching].

IPsec also has problems with NAT traversal [RFC2709] [RFC3715] [RFC3947] [RFC3948]: NAT boxes can neither examine nor modify port numbers on most IPsec-protected traffic, which causes very real problems in many environments (though not, admittedly, when protecting BGP). The net result is that IPsec usage is largely limited to virtual private network scenarios; it is rarely used or usable for individual applications over the Internet. At this point, the primary use of the existing TCP-based security method is protecting BGP sessions between routers. BGP speakers will rarely, if ever, be behind NATs, so it would seem that IPsec could be feasible in this use case. The existing TCP-MD5 is similarly hindered by the presence of NATs. The improved TCP authentication mechanism is intended for general use, not limited to BGP connections. On a per-connection configurable basis, compatibility with NATs is a goal of this work.

IPsec tunnels and ESP in transport or tunnel mode have often been criticized for their interference with firewalls and with traffic engineering, because they can hide port numbers and flags. A TCP

security option might choose to expose such fields for examination.

TLS does not suffer from any of these afflictions; however, it poses issues of its own. The integrated key management in TLS works well in many environments, but is too heavy-weight or otherwise inappropriate for others. A more serious issue is the limited scope of protection provided by TLS. It operates strictly above TCP, and thus provides no protection at all against attacks against the TCP header itself. Even if TLS is in use, it is possible for attackers to reset connections (US-CERT Advisory TA04-111A) or perpetrate other mischief that affects the TCP connection state before TLS processing occurs [I-D.ietf-tcpm-tcp-antispoof].

Since TCP-MD5 is deeply flawed and neither IPsec nor TLS currently provide the desired granularity of protection for some uses, it is clear that an intermediate protection mechanism can be justfied. There have been multiple proposals presented recently to fill this void [I-D.bonica-tcp-auth] [I-D.weis-tcp-auth-auto-ks] [I-D.touch-tcpm-tcp-simple-auth], but without an agreed-upon set of requirements, evaluating these proposals has been postponed. In Section 4 within this document, we provide a set of requirements that has been agreed upon by authors of all of the currently known proposed solutions.

4. Requirements

In this section, we present the distinguishing requirements for a future TCP security option, based on a consensus within the TCPM Authentication Option design team. These requirements are intended to be used as a means of evaluating potential solutions. These requirements partially have some basis in [RFC4808], and also have some commonality with other requirement sets developed for BGP session security [I-D.behringer-bgp-session-sec-req]. We also include some expected constraints or behaviors of a solution in Section 4.2, that are not expected to be useful in evaluating between differing approaches, but are refinements that could be compatible with any solution approach. Some suggested properties that the design team was not able to obtain a consensus for or against are listed in Appendix A.

4.1. Distinguishing Requirements

The requirements that a solution must fulfill are:

1. Protected Elements:

   A. TCP Pseudoheader

      The pseudoheader of specific IPv4 or IPv6 fields used in the computation of a segment's TCP checksum, from [RFC0793] and [RFC2460], is protected. By including source and destination IP addresses, this influences operation through NATs in a similar way to IPsec's Authentication Header, so although pseudoheader coverage MUST be possible in any viable solution, it MUST also be optional on a per-connection basis,

      For checksum purposes, the header of a TCP connection is the combination of its TCP Pseudoheader and its TCP Header. The IP addresses of the pseudoheader are included because they (together with the port numbers) define the connection; other fields are included to protect fields of the IP header that otherwise affect the TCP connection (in the latter case, largely by their inclusion in the TCP checksum).

   B. Base TCP Header

      The full base TCP header is protected, excluding any TCP options and the TCP checksum. By covering TCP port numbers, this influences operation through NATs in a similar way to IPsec's Authentication Header, so while port number coverage MUST be possible in any viable solution, it also MUST be optional on a per-connection basis.

The TCP Header is included by definition, since the purpose of this security option is to protect the TCP header.

C. TCP Options

Additionally, each defined TCP option type may be either selected for or excluded from protection. This is configured on a per-option type, per-connection basis, and is static for the lifetime of a TCP connection.

Other TCP options may or may not be protected by this security option, as desired. The primary reason for excluding options is efficiency, and because this level of protection can be relaxed in a way that impacts only an individual connection, this is a user choice.

Note that the authentication option itself MUST be included, with the authentication hash zeroed out.

D. TCP Data

The payload of each TCP segment containing the data given to applications MUST also be protected.

2. Option Structure Requirements:

A. Privacy

The authentication option MUST NOT directly expose sensitive security parameters, so that a third party's ability to view packets does not also permit them to inject authenticable packets or to otherwise determine information that could be used to compromise a particular connection, or other connections, between a pair of hosts.

B. Allow Optional

A host capable of parsing the authentication option MUST be able to requrie or ignore the option on received segments on a per-connection basis.

The purposed of the option is to authenticate connections; hosts must be able to discard segments sent to connections intended to be authenticated (i.e. they MUST be able to require the option's use). Authentication determines the ID of the source of a packet; some hosts may not be interested in verifying the ID. Presumably, use of the option would be determined a-priori, before a connection is established by a

separate key and/or policy management system, but it still may be useful to offload or otherwise ignore an expensive authentication calculation, especially if the resulting ID confirmation is not desired.

C. Require Non-Optional

A host capable of sending the authentication option MUST be able to coordinate in-band whether the option should be required or might be ignored for a particular connection with a capable receiver.

This requirement supports senders who prefer to use the option, but who are also willing to support hosts not implementing the option. Such coordination would typically happen in the key management system, but since that system could be manual, an in-band mechanism to confirm use of this option and backoff if not supported is required. This mechanism would also prevent backoff if the sender does not desire that behavior.

D. Standard Parsing

The authentication option MUST be trivially parseable by those TCP implementations that do not support it. This means that it must follow the [RFC0793] format of including a type and length field, so that it can be skipped over when it is not supported by an implementation. TCP already specifies that hosts not supporting an option ignore that option in received segments; stating this requirement here simply ensures that TCP authentication solutions do not alter the format of the base TCP header or radically depart from the typical options encoding.

E. Compatible with Large Windows

The authentication option MUST allow the concurrent use of timestamps and window-scaling within protected connections, as excluding these could limit its range of performance.

These options are in common use, and are needed for performance over high-speed or high-delay paths. Use of the authentication option thus needs to permit the use of these options, or its practical deployability will be severely limited.

F. Compatible with SACK

If the use of Selective Acknowledgements (SACK) is negotiated on a connection, the authentication option MUST allow room for at least one SACK block to be included in the TCP options, and preferably more.

This option, like (E), is in common use, and is needed for performance in large-window, lossy connections. Use of the authentication option thus needs to permit the use of SACK.

3. Cryptography Requirements

A. Baseline Defaults

There MUST be at least one set of particular cryptography algorithms or constructions whose use is supported by all implementations and can be safely assumed to be supported by any implementation of the authentication option.

This requirement is intended to support interoperability of this option, by having a single default.

B. Good Algorithms and Constructions

The authentication option MUST support default cryptography algorithms and constructions that are accepted by the community. This means it MUST NOT rely on non-standard or ad-hoc hash functions, keyed-hash constructions, signature schemes, or other functions, and MUST use published and standard schemes (i.e. it should use a construction like HMAC versus the form of keyed-hash used in TCP-MD5).

This requirement is intended to correct the flaws in the strength of authentication provided by the keyed hash used in TCP-MD5.

C. Algorithm Agility

The authentication option MUST be capable of supporting algorithms other than its defaults, in order to adapt to future discoveries. An implementation that supports multiple algorithms MUST permit concurrent connections to use different algorithms.

The existing TCP-MD5 requires substantial revision or retirement because its algorithms cannot be replaced. This requirement allows the authentication option to be agile to algorithmic attacks, where additional algorithms can be added as needed.

D. Order-Independent Processing

Authentication MUST be performed on individual, unordered TCP segments, so that it is not severely influenced by reasonable amounts of packet loss or reordering.

TCP headers are processed in the order received, although the data is reordered based on header information. As a result, header fields must be authenticated in the order received; to reorder them first would alter TCP semantics, and would potentially require data in unauthenticated segments to be quarantined (i.e. copied again) until authenticated later.

E. Parameter Changes Require Key Changes

A change in the keys used MUST accompany any change in the other parameters the cryptography functions for the authentication option are configured with.

This requirement allows the design of a compact option. It allows the key ID and key itself to indicate the parameters, rather than requiring header fields for them. It also avoids interpreting those parameters from in-band information, further avoiding exposing them to parties on the path.

4. Keying Requirements

A. Intraconnection Rekeying

Within the course of a single connection, the authentication option MUST accomodate rekeying.

TCP spoofing attacks, which this option is intended to defeat, are often targeted at relatively long-lived connections. Use of a single key over a long connection is a known security problem, so it would be preferable to either limit the length of a connection or require in-band keying support.

Unfortunately, not all applications are easy to restart. BGP, for which this option is intended, is being augmented for graceful restart [RFC4724] [RFC4781], but this extension is under recent scrutiny. TCP itself has no limit on the length of a connection, and it would be preferable to avoid modifying this semantic.

B. Efficient Rekeying

A rekeying event MUST NOT significantly affect performance of the TCP connection. Most segments should be validated by a single pass of the construction of cryptography algorithms used for authentication, and no validations should require more than a small, fixed number of passes.

Any aspect of this option which is inefficient is likely to inhibit its deployment. When using this option, segments may arrive out of order, and it would be inefficient to determine which key is appropriate via a large number of trials. Such trials would present a DoS vulnerability during rekeying. This issue is discussed in [RFC4808].

C. Automated and Manual Keying

The authentication option MUST support both manual configuration of preshared keys and automated key management. This allows for different modes of operation depending on the user's particular deployment environment.

D. Key-Mananagement Agnostic

The per-segment authentication is performed without regards to the manner in which keying material is obtained. This requirement decouples the option mechanism itself from the key management system used, so that either multiple protocols can be integrated, or any flawed methods can be easily replaced in the future.

4.2. Expected Constraints

In addition to having a wire format that supports the Distinguishing Requirements, a solution should include the following caveats in its internal operation.

1. Silent Failure

On failure (due to incorrect or missing authentication data), segments MUST be silently discarded, with no reply generated. Shuch events SHOULD be logged periodically. Failed segments MUST NOT alter the protocol state of the TCP connection itself.

Silence and the use of only periodic logging prevents the creation of a new DoS opportunity.

2. Maximum of One Option per Segment

At most, one authentication option MUST be allowed per segment.

The presence of multiple options MUST be treated as a failure.

Use of multiple options would present another DoS opportunity, and provides no additional protection vs. a single option with appropriate connection latching to other mechanisms, if desired.

3. Outgoing All-or-None Operation

Within a connection, once the authentication option is enabled, all segments MUST carry the option.

This prevents headers and/or data from being injected into a protected connection.

4. Incoming All-Checked Operation

An implementation capable of using the authentication option MUST check every incoming segment's connection state to decide whether the option's presence is required.

This requirement allows a host to determine which connections require the option, vs. which allow it as optional. Checking connection state for every incoming segment enforces required use for indicated connections.

5. Non-Interaction with TCP-MD5

An implementation MUST NOT allow a connection to simultaneously use the new authentication option and TCP-MD5. An implementation MAY support the use of either exclusively the new authentication option or exclusively TCP-MD5 for each individual connection.

This option is intended to supercede TCP-MD5, and in the spirit of (2) above, only one such option is useful per connection. Support for existing TCP-MD5 would support legacy interoperation.

6. Optional ICMP Discard

An implementation MUST be configurable to allow a protected connection to ignore incoming ICMP Type 3 messages with Codes 2-4. This SHOULD be the default configuration.

This requirement prevents an ICMP attack on protected connections via unprotected/unauthenticable (ICMP) packets.

5. Security Considerations

This document does not specify any protocol; it discusses known security problems with a currently deployed protocol, and the requirements for fixing those problems in a new protocol. This document is itself a set of security considerations, and its publication raises no new security considerations.

6. IANA Considerations

This document does not update or create any IANA registries.

7. Informative References

[I-D.behringer-bgp-session-sec-req]
Behringer, M., "BGP Session Security Requirements", draft-behringer-bgp-session-sec-req-01 (work in progress), May 2007.

[I-D.bellovin-useipsec]
Bellovin, S., "Guidelines for Mandating the Use of IPsec", draft-bellovin-useipsec-06 (work in progress), February 2007.

[I-D.bonica-tcp-auth]
Bonica, R., "Authentication for TCP-based Routing and Management Protocols", draft-bonica-tcp-auth-06 (work in progress), February 2007.

[I-D.ietf-btns-connection-latching]
Williams, N., "IPsec Channels: Connection Latching", draft-ietf-btns-connection-latching-01 (work in progress), March 2007.

[I-D.ietf-tcpm-tcp-antispoof]
Touch, J., "Defending TCP Against Spoofing Attacks", draft-ietf-tcpm-tcp-antispoof-06 (work in progress), February 2007.

[I-D.ietf-tcpm-tcpsecure]
Ramaiah, A., "Improving TCP's Robustness to Blind In-Window Attacks", draft-ietf-tcpm-tcpsecure-07 (work in progress), February 2007.

[I-D.manral-rpsec-existing-crypto]
Manral, V., "Issues with existing Cryptographic Protection Methods for Routing Protocols", draft-manral-rpsec-existing-crypto-04 (work in progress), April 2007.

[I-D.touch-tcpm-tcp-simple-auth]
Touch, J. and A. Mankin, "The TCP Simple Authentication Option", Internet-Draft draft-touch-tcpm-tcp-simple-auth-02 (work in progress),, October 2006.

[I-D.weis-tcp-auth-auto-ks]
Weis, B., "Automated key selection extension for the TCP Enhanced Authentication Option", draft-weis-tcp-auth-auto-ks-02 (work in progress),

March 2007.

[RFC0793] Postel, J., "Transmission Control Protocol", STD 7, RFC 793, September 1981.

[RFC1810] Touch, J., "Report on MD5 Performance", RFC 1810, June 1995.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2385] Heffernan, A., "Protection of BGP Sessions via the TCP MD5 Signature Option", RFC 2385, August 1998.

[RFC2460] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", RFC 2460, December 1998.

[RFC2709] Srisuresh, P., "Security Model with Tunnel-mode IPsec for NAT Domains", RFC 2709, October 1999.

[RFC3562] Leech, M., "Key Management Considerations for the TCP MD5 Signature Option", RFC 3562, July 2003.

[RFC3715] Aboba, B. and W. Dixon, "IPsec-Network Address Translation (NAT) Compatibility Requirements", RFC 3715, March 2004.

[RFC3947] Kivinen, T., Swander, B., Huttunen, A., and V. Volpe, "Negotiation of NAT-Traversal in the IKE", RFC 3947, January 2005.

[RFC3948] Huttunen, A., Swander, B., Volpe, V., DiBurro, L., and M. Stenberg, "UDP Encapsulation of IPsec ESP Packets", RFC 3948, January 2005.

[RFC4107] Bellovin, S. and R. Housley, "Guidelines for Cryptographic Key Management", BCP 107, RFC 4107, June 2005.

[RFC4278] Bellovin, S. and A. Zinin, "Standards Maturity Variance Regarding the TCP MD5 Signature Option (RFC 2385) and the BGP-4 Specification", RFC 4278, January 2006.

[RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, December 2005.

[RFC4346] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.1", RFC 4346, April 2006.

[RFC4724] Sangli, S., Chen, E., Fernando, R., Scudder, J., and Y.

Rekhter, "Graceful Restart Mechanism for BGP", RFC 4724, January 2007.

[RFC4781] Rekhter, Y. and R. Aggarwal, "Graceful Restart Mechanism for BGP with MPLS", RFC 4781, January 2007.

[RFC4808] Bellovin, S., "Key Change Strategies for TCP-MD5", RFC 4808, March 2007.

[SP4] Dinkel, C., "Secure Data Network System (SDNS) Network, Transport, and Message Security Protocols", NISTIR 90-4250, 1990.

[SP4P] Branstad, D., Dorman, J., Housley, R., and J. Randall, "SP4: A Transport Encapsulation Security Protocol", Third Aerospace Security Conference Proceedings, December 1987.

[TLSP] "Information Technology -- Telecommunications and Information Exchange Between systems -- Transport Layer Security Protocol", ISO/IEC 10736, 1995.

Appendix A. Un-Agreed Properties

There were some items that were suggested as requirements but which were not ratified by all participants in the design team. These are listed here.

1. Saves Work When Optional

   A host sending TCP segments should be able to detect on a per-connection basis whether the authentication option is required or is being ignored by a receiver who supports the option.

2. Single-Pass Rekeying

   The authentication option should support rekeying where incoming segments are validated using a single pass of the cryptographic construction used for authentication.

Authors' Addresses

Wesley M. Eddy (editor)
Verizon Federal Network Systems
NASA Glenn Research Center
21000 Brookpark Rd, MS 54-5
Cleveland, OH 44135

Phone: 216-433-6682
Email: weddy@grc.nasa.gov

Steven M. Bellovin
Columbia University
1214 Amsterdam Avenue
MC 0401
New York, NY 10027

Phone: +1 212 939 7149
Email: bellovin@acm.org

Joe Touch
USC/ISI
4676 Admirality Way
Marina del Rey, CA 90292-6695

Phone: +1 (310) 448-9151
Email: touch@isi.edu

Ronald P. Bonica
Juniper Networks
2251 Corporate Park Drive
Herndon, VA 20171

Email: rbonica@juniper.net

Full Copyright Statement

Intellectual Property

Acknowledgment

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