INTERNET-DRAFT
Intended status: Experimental

W A Simpson
DayDreamer

4 July 2011

TCP Cookie Transactions (TCPCT) Rapid Restart draft-simpson-tcpct-rr-02

Abstract

TCP Cookie Transactions (TCPCT) [RFC6013] deter spoofing of connections and prevent resource exhaustion, eliminating Responder (server) state during the initial handshake. The Initiator (client) has sole responsibility for ensuring required delays between connections. The cookie exchange may carry data, limited to inhibit amplification and reflection denial of service attacks.

This specification provides an optional rapid restart facility for persistent connections.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current.

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."

Applicability

This specification is intended for network paths under the complete control of an operator, such as secure tunnels or intra-campus private links. Widely deployed security firewalls block the transmission of these additional data segments, and are outside the scope of this specification.

Table of Contents

1. Introduction

TCP Cookie Transactions (TCPCT) [RFC6013] provide a cryptologically secure mechanism to guard against simple flooding attacks sent with bogus IP [RFC791] Sources or TCP [RFC793] Ports.

Also, implementations may optionally exchange limited amounts of transaction data during the initial cookie exchange, reducing network latency and host task context switching.

This optional facility allows additional data segments for second and subsequent cookie transactions, immediately following the Responder's <SYN,ACK(SYN)> and prior to receipt of the Initiator's <ACK(SYN)>. The amount of data is limited by both the Initiator's advertised window (rwnd), and the Responder's vestigial congestion window (VCW) calculated by timestamps retained from the previous connection.

Where repeated transactions are initiated within 1/2 the Round Trip Time (RTT), assuming symmetrical paths, the entire congestion window (cwnd) remains open. Most efficacious for persistent connections over long-delay paths.

1.1. Terminology

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

byte An 8-bit quantity; also known as "octet" in standardese.

2. Protocol Overview

This optional facility consist of several simple phases following the initial TCPCT connection. (See [RFC6013] Protocol Overview for previous steps.)

3. During close (or reset) of the TCP connection, the Timestamps and Cookie-Pair options guard the exchange.

   If the Responder (server) application has set the TCPCT_RETAIN flag prior to the close (or reset) of the connection, the Responder retains its Transport Control Block (TCB) [RFC793] for a limited time.

4. If the Initiator (client) application has set the TCPCT_RETAIN flag, rather than expunging the TCB after TIME-WAIT, the implementation retains its TCB indefinitely.

   When a retained TCB exists, the Responder MAY send additional data segments.

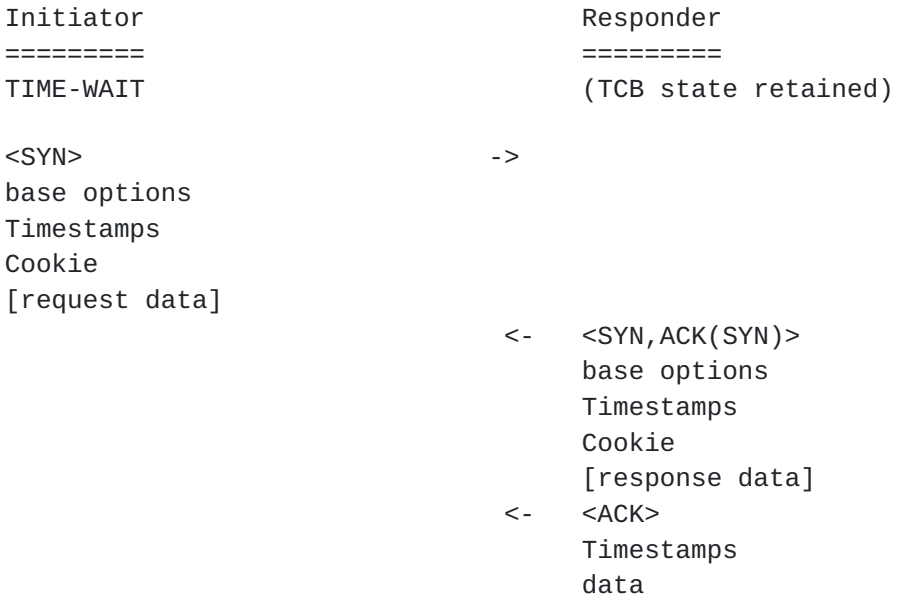
5. The Initiator sends its <ACK(SYN)> with Timestamps Extended Option and Cookie-Pair, optionally with Selective Acknowledgment (Sack) [RFC2018] to acknowledge any <SYN,ACK(SYN)> data.

   However, the Initiator does not acknowledge any additional data segments until receipt of the corresponding Responder <ACK> (or <FIN,ACK>) with Timestamps Extended Option and Cookie-Pair.

6. Connection closes as described in 3. Repeat process for each reconnection.

The sequence of messages is summarized in the diagram below.

2.1. Message Summary (Simplified)



```
   <FIN,ACK(SYN)>                  ->
   full options
   Timestamps
   Cookie-Pair
   [Sack(response)]
                                    <-   <FIN,ACK(FIN)>
                                         full options
                                         Timestamps
                                         Cookie-Pair
                                         data
                                         (TCB state retained)

   <SYN>                           ->
   base options
   Timestamps
   Cookie
   [request data]
                                    <-   <SYN,ACK(SYN)>
                                         base options
                                         Timestamps
                                         Cookie
                                         [response data]
                                    <-   <ACK>
                                         Timestamps
                                         data

   <ACK(SYN)>                      ->
   full options
   Timestamps
   Cookie-Pair
   [Sack(response)]
   data
                                    <-   <FIN,ACK>
                                         full options
                                         Timestamps
                                         Cookie-Pair
                                         data
```

```
<FIN,ACK(FIN)>                ->
Timestamps
Cookie-Pair
data
                                <-   <ACK(FIN)>
                                     Timestamps
                                     Cookie-Pair
                                     (TCB state retained)
TIME-WAIT
```

The upper transaction illustrates a single segment accelerated open, rapid restart, accelerated Initiator close, and advisory <FIN>.

The lower transaction illustrates a multiple segment accelerated open, rapid restart, and normal Responder close.

3. Protocol Details

Support for rapid restart is OPTIONAL, and depends upon support for [RFC6013] Accelerated Open. If the symbol TCPCT_RETAIN is defined in the system headers provided at the time of compilation, the implementation supports rapid restart.

Although the Initiator is expected to reuse the same TCP Source Port, intervening middleboxes [RFC3234] are likely to expire any [RFC3022] port translations during the time between connections. Instead, the IP Source, Initiator Cookie, and Timestamps Echo Reply fields are checked against the retained TCB of prior connections.

3.1. Responder TCB Retention

By default, upon receipt of the Initiator <ACK(FIN)> (and verification of the Timestamps and Cookie-Pair options), the Responder removes its TCB.

If the Responder (server) application has set the TCPCT_RETAIN flag, the Responder calculates the retention time. This time is based on the anticipated reduction of the congestion window during a short idle period. For purposes of these calculations, the implementation SHOULD reduce its congestion window by half for every Round Trip Time (RTT) that the flow has remained idle. [RFC2861]

Where repeated transactions are initiated within 1/2 the Round Trip Time (RTT), assuming symmetrical paths, the entire congestion window (cwnd) remains open.

This approach yields advantage for even a vestigial congestion window (VCW) less than the [RFC5681] Initial Window (IW). Unlike the [RFC5681] Restart Window (RW), VCW is not subject to the retransmission timeout (RTO).

When VCW is less than or equal to TCP_SYN_ACK_DATA_LIMIT (or the local value in TCPCT_S_DATA_DESIRED) plus Maximum Segment Size (MSS), as limited by the retained Path Maximum Transmission Unit (PMTU), rapid restart offers no improvement over accelerated open. At that time, the Responder removes its TCB.

3.2. Initiator <SYN> Data

By default, the Initiator <SYN> does not contain data. The application sets TCPCT_S_DATA_DESIRED to indicate that the <SYN> MAY be sent with data.

The Initiator uses the existing Initiator Cookie and fills the Timestamps Echo Reply field with the least significant 32 bits of the most recent Responder Timestamps Value. Any existing TSoffset MUST be incremented.

During the rapid restart exchange, the Initiator is solely responsible for retransmission.

3.3. Responder <SYN,ACK(SYN)> Data

By default, the Responder <SYN,ACK(SYN)> does not contain data. The application sets TCPCT_S_DATA_DESIRED to indicate that the <SYN,ACK(SYN)> MAY be sent with data.

Upon receipt of the <SYN> with a Cookie option, the Responder MAY process any data present. If the initial data is not accepted, the Acknowledgment Number will be the received Sequence Number plus one (1) for the <SYN>.

When a retained TCB exists, the Responder compares the IP Source, Initiator Cookie, and Timestamps Echo Reply. If the corresponding fields exactly match the most recent values, the Responder MAY send additional data segments. This segment data is limited to the retained Path Maximum Transmission Unit (PMTU).

The Responder updates the TCB Source Port, and recalculates its Timestamps Value. Any existing TSoffset MUST be incremented. The same Timestamps fields are used for all data segments. As the Timestamps Extended Option has not been received, the standard

[RFC1323] (32-bit) Timestamps option is sent.

If the segment data is the entire response (there is no further data expected), the Responder MUST NOT send the final segment and <FIN> MUST NOT be set.

Although the Responder retains TCB state, retransmission timers are not used. Arrival of an Initiator's retransmission appears to be an original <SYN> transmission.

As the Responder's Timestamps Value has been recalculated, these subsequent connection attempts MUST NOT trigger rapid restart. This will inhibit self-inflicted Denial of Service (DoS), and prevent spoofed amplification and reflection attacks [RFC5358].

3.4. Initiator <ACK(SYN)> Data

Upon receipt of the <SYN,ACK(SYN)> with a Cookie option, the Initiator MUST process any data present. In this case, the internal RCV.NXT is advanced to provide at-most-once semantics.

If the Selective Acknowledgment (Sack) option [RFC2018] has been successfully negotiated, a short Sack acknowledging the response data MUST be sent following the Cookie-Pair in the extended header.

At this time, additional segments MAY be sent, according to the usual [RFC5681] TCP congestion control process.

However, the Initiator MUST NOT acknowledge any additional data segments, until receipt of the corresponding Responder <ACK> (or <FIN,ACK>) with Timestamps Extended Option and Cookie-Pair. Upon retransmission of the <ACK(SYN)>, any Sack SHOULD include the accumulated unacknowledged bytes.

3.5. Responder <ACK> Data

Upon receipt of the <ACK(SYN)> with a Cookie-Pair option (and verification of the Timestamps and Cookie-Pair options), the Responder SHOULD process any data present.

Since the TCP Sequence and Acknowledgment Numbers have not advanced, the Responder will process the same incoming data, and generate the same response.

If VCW is less than IW, reset cwnd to IW. Then, increase cwnd by the number (L) of previously unacknowledged bytes indicated by any

incoming Sack (similar to [RFC3465]).

cwnd = max(IW, VCW) + L

At this time, additional segments MAY be sent, according to the usual [RFC5681] TCP congestion control process.

Acknowledgments

Yuchung Cheng, H. K. Jerry Chu, Sivasankar Radhakrishnan, and Arvind Jain described exchanging a token for "fast open" on subsequent connections. [CCRJ2011] That feature was subsumed by this specification.

Many thanks to Mark Allman, Wesley Eddy, Richard Scheffenegger, and Paul Vixie for helpful comments.

IANA Considerations

This document has no IANA actions.

[RFC Editor: please remove this section prior to publication.]

Operational Considerations

Any implementation of this specification SHOULD be configurable, separately for each port or connection.

TCPCT_RETAIN
: When this symbol is defined in the system headers provided at the time of compilation, the optional TCPCT Rapid Restart feature is available.

  Default: 0 (off). Indicates TCPCT TCB SHOULD be retained for rapid restart.

TCPCT_S_DATA_DESIRED
: Default: 0. The maximum amount of data transmitted with the <SYN> (up to TCP_SYN_DATA_LIMIT) or the <SYN,ACK(SYN)> (up to TCP_SYN_ACK_DATA_LIMIT).

  Whenever this field is non-zero, wait for data before sending. Unlike TCPCT_COOKIE_DESIRED, this field MUST be set explicitly; there is no system value.

Security Considerations

TCPCT was based on currently available tools, by experienced network protocol designers with an interest in cryptography, rather than by cryptographers with an interest in network protocols. This specification is intended to be readily implementable without requiring an extensive background in cryptology.

Therefore, only minimal background cryptologic discussion and rationale is included in this document. Although some review has been provided by the general cryptologic community, it is anticipated that design decisions and tradeoffs will be thoroughly analysed in subsequent dissertations and debated for many years to come. Cryptologic details are reserved for separate documents that may be more readily and timely updated with new analysis.

The security depends on the quality of the random numbers generated by each party. Generating cryptographic quality random numbers on a general purpose computer without hardware assistance is a very tricky problem (see [RFC4086] for discussion).

TCPCT is not intended to prevent or recover from all possible security threats. Rather, it is designed to inhibit inadvertent middlebox interference, while protecting against Denial of Service (DoS) attacks. (See [RFC4732], and [RFC3552] section 4.6.3 et seq.)

The cookie exchange does not protect against an interloper that can race to substitute another value, nor an interceptor that can modify and/or replace a value. These attacks are considerably more difficult than passive vacuum-cleaner monitoring.

The initial exchange is most fragile, as protection against spoofing relies entirely upon the sequence and timestamp. Instead, the IP Source, Initiator Cookie, and Timestamps Echo Reply fields are checked against the retained TCB of prior connections. This is considerably stronger than the initial exchange; in effect, a partial extension of the three-way handshake closing the prior connection.

Normative References

[RFC791] Postel, J., "Internet Protocol", STD 5, September 1981.

[RFC793] Postel, J., "Transmission Control Protocol", STD 7, September 1981.

[RFC1323] Jacobson, V., Braden, R., and D. Borman, "TCP Extensions for High Performance", May 1992.

[RFC2018] Mathis, M., Mahdavi, J., Floyd, S., and A. Romanow, "TCP Selective Acknowledgment Options", October 1996.

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

[RFC5681] Allman, M., Paxson, V., and E. Blanton, "TCP Congestion Control", September 2009.

[RFC6013] Simpson, W. A., "TCP Cookie Transactions (TCPCT)", January 2011.

Informative References

[CCRJ2011] Cheng, Y., Chu, J., Radhakrishnan, S., and A. Jain, "TCP Fast Open", work in progress, March 7, 2011. http://tools.ietf.org/html/draft-cheng-tcpm-fastopen

[RFC2861] Handley, M., Padhye, J., and S. Floyd, "TCP Congestion Window Validation", June 2000.

[RFC3022] Srisuresh, P., and K. Egevang, "Traditional IP Network Address Translator (Traditional NAT)", January 2001.

[RFC3234] Carpenter, B., and S. Brim, "Middleboxes: Taxonomy and Issues", February 2002.

[RFC3465] Allman, M., "TCP Congestion Control with Appropriate Byte Counting (ABC)", February 2003.

[RFC3552] Rescorla, E., and B. Korver, "Guidelines for Writing RFC Text on Security Considerations", BCP 72, July 2003.

[RFC4086] Eastlake, D. (3rd), Schiller, J., and S. Crocker, "Randomness Requirements for Security", BCP 106, June 2005.

[RFC4732] Handley, M., Ed., and Rescorla, E., Ed., and Internet Architecture Board, "Internet Denial-of-Service Considerations", November 2006.

[RFC4987] Eddy, W., "TCP SYN Flooding Attacks and Common Mitigations", August 2007.

[RFC5077] Salowey, J., Zhou, H., Eronen, P., and H. Tschofenig, "Transport Layer Security (TLS) Session Resumption without Server-Side State", January 2008.

[RFC5358] Damas, J., and F. Neves, "Preventing Use of Recursive Nameservers in Reflector Attacks", BCP 140, October 2008.

Author's Address

Questions about this document can be directed to:

William Allen Simpson
DayDreamer
Computer Systems Consulting Services
1384 Fontaine
Madison Heights, Michigan 48071

William.Allen.Simpson@Gmail.com