TLS Working Group
Internet-Draft
Intended status: Standards Track
Expires: June 08, 2014

N. Mathewson
Tor
B. Laurie
Google
December 05, 2013

Deprecating gmt_unix_time in TLS
draft-mathewson-no-gmtunixtime-00

Abstract

This memo deprecates the use of the gmt_unix_time field for sending the current time in all versions of the TLS protocol's handshake. A rationale is provided for this decision, and alternatives are discussed.

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."

This Internet-Draft will expire on June 08, 2014.

Copyright Notice

Table of Contents

1. Introduction 2
1.1. The current behavior of gmt_unix_time 2
1.2. gmt_unix_time is an undesirable client fingerprint . . . 2
1.3. gmt_unix_time is undesirable on servers as well 3
1.4. gmt_unix_time is neither adequate nor necessary for its intended purpose 3
2. Deprecating gmt_unix_time 4
3. Security considerations 4
4. References 4
4.1. Normative References 4
4.2. Informative References 5
Authors' Addresses 5

1. Introduction

Current versions of the TLS protocol, dating back to the SSL 3.0, describe a gmt_unix_time field, sent in the clear, as part of the TLS handshake. While the exact format of this field is not strictly specified, typical implementations fill it with the time since the Unix epoch (Jan 1, 1970) in seconds. This practice is neither necessary nor safe.

1.1. The current behavior of gmt_unix_time

According to RFC 2246 ("The TLS Protocol Version 1.0"), gmt_unix_time holds "The current time and date in standard UNIX 32-bit format (seconds since the midnight starting Jan 1, 1970, GMT) according to the sender's internal clock. Clocks are not required to be set correctly by the basic TLS Protocol; higher level or application protocols may define additional requirements." This text is retained unchanged in RFC 4346 and in RFC 5246.

The gmt_unix_time field was first introduced in SSL 3.0, the predecessor to TLS 1.0. The field was meant to preserve the protocol's robustness in the presence of unreliable random number generators that might generate the same random values more than once. If this happened, then SSL would be vulnerable to various attacks based on the non-uniqueness of the Random fields in the ClientHello and ServerHello messages. Using the time value in this way was meant to prevent the same Random value from being sent more than once, even in the presence of misbehaved random number generators.

1.2. gmt_unix_time is an undesirable client fingerprint

Despite the late date, much of the world is still not synchronized to the second via an ntp-like service. This means that different

clients have different views of the current time. By declaring their view of the time in the gmt_unix_time field, clients provide eavesdroppers a fingerprint that helps to track and distinguish them. This fingerprint is useful for tracking mobile clients as they move around from network to network. It can also distinguish clients who would otherwise be anonymized by using a VPN, NAT, or privacy network.

Even when clocks are synchronized to within a second, an eavesdropper who can observe multiple gmt_unix_time values over time can build a statistical profile of when within a single second a client transitions from one second to another, and thereby distinguish such clients.

Finally, an active packet-forging attacker who can forge NTP replies to a targeted client can introduce anomalies in that client's view of the current time. By (say) slowly advancing a client a fixed interval into the future, the attacker can set a time-based plaintext "cookie" that will persist on the user's TLS connections for so long as the user's view of the current time remains skewed. (The system of RFC 5906 prevents this attack by authenticating NTP replies, but it is not universally used.)

1.3. gmt_unix_time is undesirable on servers as well

While some protocol designs retain a clear separation between (nominally anonymous, possibly privacy-seeking) clients, and (well known, easy to find) servers, there are several counterexamples in which the responder to a TLS connection (the "Server" in the language of the TLS specification) also wishes to avoid fingerprinting. These include services provided through an anonymizing service, and participation in some peer-to-peer network designs.

1.4. gmt_unix_time is neither adequate nor necessary for its intended purpose

One might argue that the problems discussed above are real, but that the benefit of the gmt_unix_time field outweighs them. But in fact, the field provides no actual benefit.

The purpose of gmt_unix_time is to render repeated PRNG output values survivable. But other portions of the TLS protocol -- for example, the ephemeral key ciphersuites -- remove this alleged benefit. If an implementation is prone to repeating the same ClientHello.Random values when it starts up, it is also likely prone to repeating those same values as Diffie-Hellman private keys, thereby rendering its connections insecure.

Even if the ephemeral key ciphersuites are not in use, it's not unusual in practice on a modern computer for multiple TLS clients or servers to initiate handshakes around the same second. A one-second time value is insufficient to ensure uniqueness.

Further, even if gmt_unix_time were adequate, it would not be necessary. Given an even minimally cryptographically adequate PRNG, and a non-repeating clock, implementors can ensure that the PRNG generates non-repeated values by adding the current (high resolution) time to the PRNG state when seeding it. This is not enough to ensure that the PRNG is unpredictable by an attacker, but it does ensure that the PRNG will not generate duplicate values even in the presence of inadequate entropy sources. Most cryptographic libraries and operating system PRNGs take this approach today.

2. Deprecating gmt_unix_time

For the reasons we discuss above, we recommend that TLS implementors MUST by default set the entire value the ClientHello.Random and ServerHello.Random fields, including gmt_unix_time, to a cryptographically random sequence.

If existing users of a TLS implementation may rely on gmt_unix_time containing the current time, we recommend that implementors MAY provide the ability to set gmt_unix_time as an option only, off by default.

Generating cryptographically strong pseudorandom sequences is beyond the scope of this document. Nevertheless, we recommend that implementors who may be concerned about the loss of the benefits of the gmt_unix_time field should ensure that their cryptographic PRNG input includes -- among the entropy that they hope will be strong -- a high-resolution view of the current time.

3. Security considerations

The entire document is security-related.

4. References

4.1. Normative References

[RFC2246] Dierks, T. and C. Allen, "The TLS Protocol Version 1.0", RFC 2246, January 1999.

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

[RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, August 2008.

4.2. Informative References

[RFC4086] Eastlake, D., Schiller, J., and S. Crocker, "Randomness Requirements for Security", BCP 106, RFC 4086, June 2005.

[RFC5905] Mills, D., Martin, J., Burbank, J., and W. Kasch, "Network Time Protocol Version 4: Protocol and Algorithms Specification", RFC 5905, June 2010.

[RFC5906] Haberman, B. and D. Mills, "Network Time Protocol Version 4: Autokey Specification", RFC 5906, June 2010.

Authors' Addresses

Nick Mathewson
The Tor Project

Email: nickm@torproject.org

Ben Laurie
Google

Email: benl@google.com