Recommendations for Secure Use of TLS and DTLS
draft-ietf-uta-tls-bcp-04

Abstract

   Transport Layer Security (TLS) and Datagram Transport Security Layer
   (DTLS) are widely used to protect data exchanged over application
   protocols such as HTTP, SMTP, IMAP, POP, SIP, and XMPP. Over the
   last few years, several serious attacks on TLS have emerged,
   including attacks on its most commonly used cipher suites and modes
   of operation. This document provides recommendations for improving
   the security of deployed services that use TLS and DTLS. The
   recommendations are applicable to the majority of use cases.

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 April 3, 2015.

Copyright Notice

Table of Contents

1. Introduction

   Transport Layer Security (TLS) and Datagram Transport Security Layer
   (DTLS) are widely used to protect data exchanged over application
   protocols such as HTTP, SMTP, IMAP, POP, SIP, and XMPP. Over the
   last few years, several serious attacks on TLS have emerged,
   including attacks on its most commonly used cipher suites and modes
   of operation. For instance, both AES-CBC and RC4, which together
   comprise most current usage, have been attacked in the context of
   TLS. A companion document [I-D.ietf-uta-tls-attacks] provides
   detailed information about these attacks.

   Because of these attacks, those who implement and deploy TLS and DTLS
   need updated guidance on how TLS can be used securely. Note that
   this document provides guidance for deployed services, as well as
   software implementations, assuming the implementer expects his or her
   code to be deployed in environments defined in the following section.
   In fact, this document calls for the deployment of algorithms that
   are widely implemented but not yet widely deployed. Concerning
   deployment, this document targets a wide audience, namely all
   deployers who wish to add confidentiality and data integrity
   protection to their communications. In many (but not all) cases
   authentication is also desired. This document does not address the
   rare deployment scenarios where no confidentiality is desired.

   The recommendations herein take into consideration the security of
   various mechanisms, their technical maturity and interoperability,
   and their prevalence in implementations at the time of writing.
   Unless noted otherwise, these recommendations apply to both TLS and
   DTLS. TLS 1.3, when it is standardized and deployed in the field,
   should resolve the current vulnerabilities while providing
   significantly better functionality, and will very likely obsolete
   this document.

   These are minimum recommendations for the use of TLS for the
   specified audience. Individual specifications may have stricter
   requirements related to one or more aspects of the protocol, based on
   their particular circumstances. When that is the case, implementers
   MUST adhere to those stricter requirements.

   Community knowledge about the strength of various algorithms and
   feasible attacks can change quickly, and experience shows that a
   security BCP is a point-in-time statement. Readers are advised to
   seek out any errata or updates that apply to this document.

2. **Intended Audience and Applicability Statement**

   In the following, we specify which audience this document addresses
   concerning deployment. This document applies only to environments
   where confidentiality is required. It recommends algorithms and
   configuration options that make secrecy of the data-in-transit
   mandatory. While this includes the majority of the TLS use cases,
   there are some notable exceptions.

   This document assumes that data integrity protection is always one of
   the goals of a deployment. In cases when integrity is not required,
   it does not make sense to employ TLS in the first place. There are
   attacks against confidentiality-only protection that utilize the lack
   of integrity to also break confidentiality (see e.g. [DegabrieleP07]
   in the context of IPsec). Thus, even when using opportunistic
   encryption, it is essential to provide cryptographic data integrity
   protection

2.1. **Security Services**

   This document provides recommendations for an audience that wishes to
   secure their communication with TLS to achieve the following:

   o Confidentiality: all (payload) communication is encrypted with the
      goal that no party should be able to decrypt it except the
      intended receiver.

   o Data integrity: any changes made to the communication are
      detectable by the receiver.

   o Optionally, authentication: this means that an end-point of the
      TLS communication is authenticated as the intended entity to
      communicate with. TLS allows to authenticate one or both end-
      points in the communication.

   Deployers MUST verify that they do not need one of the above security
   services if they deviate from the recommendations given in this
   document.

2.2. **Examples**

   The intended audience covers those services that are most commonly
   used on the Internet. Typically, all communication between clients
   and servers requires all three of the above security services.

   o Operators of WWW servers (HTTPS).

   o Operators of email servers who wish to protect the application-
      layer protocols with TLS (e.g., IMAP, POP3, or SMTP between client
      and server).

   o Operators of instant-messaging services who wish to protect their
      application-layer protocols with TLS (e.g. XMPP or IRC between
      client and server).

   An example of an audience not needing confidentiality is the
   following: a monitored network where the authorities in charge of
   that traffic domain require full access to unencrypted (plaintext)
   traffic, and where users collaborate and send their traffic in the
   clear.

3. Conventions used in this document

   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].

4. General Recommendations

   This section provides general recommendations on the secure use of
   TLS. Recommendations related to cipher suites are discussed in the
   following section.

4.1. Protocol Versions

   It is important both to stop using old, less secure versions of SSL/
   TLS and to start using modern, more secure versions. Therefore:

   o Implementations MUST NOT negotiate SSL version 2.

      Rationale: SSLv2 is considered today as insecure [RFC6176].

   o Implementations MUST NOT negotiate SSL version 3.

      Rationale: SSLv3 [RFC6101] was an improvement over SSLv2 and
      plugged some significant security holes, but did not support
      strong cipher suites. In addition, SSLv3 does not support TLS
      extensions, some of which are considered security-critical today.

   o Implementations SHOULD NOT negotiate TLS version 1.0 [RFC2246].

      Rationale: TLS 1.0 (published in 1999) does not support many
      modern, strong cipher suites.

   o Implementations MAY negotiate TLS version 1.1 [RFC4346].

Rationale: TLS 1.1 (published in 2006) is a security improvement
over TLS 1.0, but still does not support certain stronger cipher
suites.

o Implementations MUST support, and prefer to negotiate, TLS version
   1.2 [RFC5246].

   Rationale: Several stronger cipher suites are available only with
   TLS 1.2 (published in 2008).

This BCP applies to TLS 1.2. It is not safe for readers to assume
that the recommendations in this BCP apply to any future version of
TLS.

4.2. Applicability to DTLS

DTLS [RFC4347] [RFC6347] is an adaptation of TLS for UDP datagrams.

With respect to the recommendations in the current document, DTLS 1.0
is equivalent to TLS 1.1. The only exception is RC4 which is
disallowed in DTLS. DTLS 1.2 is equivalent to TLS 1.2.

4.3. Fallback to SSL

Some client implementations revert to lower versions of TLS or even
to SSLv3 if the server rejected higher versions of the protocol.
This fall back can be forced by a man in the middle (MITM) attacker.
By default, such clients MUST NOT fall back to SSLv3.

Rationale: TLS 1.0 and SSLv3 are significantly less secure than TLS
1.2, the version recommended by this document. While TLS 1.0-only
servers are still quite common, IP scans show that SSLv3-only servers
amount to only about 3% of the current Web server population.

4.4. Strict TLS

Combining unprotected and TLS-protected communication opens the way
to SSL Stripping and similar attacks. Therefore:

o In cases where an application protocol allows implementations or
   deployments a choice between strict TLS configuration and dynamic
   upgrade from unencrypted to TLS-protected traffic (such as
   STARTTLS), clients and servers SHOULD prefer strict TLS
   configuration.

o HTTP client and server implementations MUST support the HTTP
   Strict Transport Security (HSTS) header [RFC6797], in order to

allow Web servers to advertise that they are willing to accept
TLS-only clients.

o When applicable, Web servers SHOULD use HSTS to indicate that they
   are willing to accept TLS-only clients.

4.5. Compression

Implementations and deployments SHOULD disable TLS-level compression
([RFC5246], Sec. 6.2.2), because it has been subject to security
attacks.

Implementers should note that compression at higher protocol levels
can allow an active attacker to extract cleartext information from
the connection. The BREACH attack is one such case. These issues
can only be mitigated outside of TLS and are thus out of scope of the
current document. See Sec. 2.5 of [I-D.ietf-uta-tls-attacks] for
further details.

4.6. TLS Session Resumption

If TLS session resumption is used, care ought to be taken to do so
safely. In particular, when using session tickets [RFC5077], the
resumption information MUST be authenticated and encrypted to prevent
modification or eavesdropping by an attacker. Further
recommendations apply to session tickets:

o A strong cipher suite MUST be used when encrypting the ticket (as
   least as strong as the main TLS cipher suite).

o Ticket keys MUST be changed regularly, e.g. once every week, so as
   not to negate the benefits of forward secrecy (see Section 7.3 for
   details on forward secrecy).

o Session ticket validity SHOULD be limited to a reasonable duration
   (e.g. 1 day), for similar reasons.

4.7. TLS Renegotiation

Where handshake renegotiation is implemented, both clients and
servers MUST implement the renegotiation_info extension, as defined
in [RFC5746].

To counter the Triple Handshake attack, we adopt the recommendation
from [triple-handshake]: TLS clients SHOULD ensure that all
certificates received over a connection are valid for the current
server endpoint, and abort the handshake if they are not. In some

usages, it may be simplest to refuse any change of certificates
during renegotiation.

4.8. Server Name Indication

TLS implementations MUST support the Server Name Indication (SNI)
extension for those higher level protocols which would benefit from
it, including HTTPS. However, unlike implementation, the use of SNI
in particular circumstances is a matter of local policy.

5. Recommendations: Cipher Suites

TLS and its implementations provide considerable flexibility in the
selection of cipher suites. Unfortunately many available cipher
suites are insecure, and so misconfiguration can easily result in
reduced security. This section includes recommendations on the
selection and negotiation of cipher suites.

5.1. General Guidelines

It is important both to stop using old, insecure cipher suites and to
start using modern, more secure cipher suites. Therefore:

o Implementations MUST NOT negotiate the NULL cipher suites.

   Rationale: The NULL cipher suites offer no encryption whatsoever
   and thus are completely insecure.

o Implementations MUST NOT negotiate RC4 cipher suites

   Rationale: The RC4 stream cipher has a variety of cryptographic
   weaknesses, as documented in [I-D.ietf-tls-prohibiting-rc4].

o Implementations MUST NOT negotiate cipher suites offering only so-
   called "export-level" encryption (including algorithms with 40
   bits or 56 bits of security).

   Rationale: These cipher suites are deliberately "dumbed down" and
   are very easy to break.

o Applications MUST NOT negotiate cipher suites of less than 112
   bits of security.

o Implementations SHOULD NOT negotiate cipher suites that use
   algorithms offering less than 128 bits of security. Note that
   some legacy cipher suites (e.g. 168-bit 3DES) have an effective
   key length which is smaller than their nominal key length (112

bits in the case of 3DES). Such cipher suites should be evaluated
according to their effective key length.

Rationale: Although these cipher suites are not actively subject
to breakage, their useful lifespan is short enough that stronger
cipher suites are desirable. 128-bit ciphers are expected to
remain secure for at least several years, and 256-bit ciphers
"until the next fundamental technology breakthrough".

o Implementations MUST support, and SHOULD prefer to negotiate,
cipher suites offering forward secrecy, such as those in the
Ephemeral Diffie-Hellman and Elliptic Curve Ephemeral Diffie
Hellman ("DHE" and "ECDHE") families.

Rationale: Forward secrecy (sometimes called "perfect forward
secrecy") prevents the recovery of information that was encrypted
with older session keys, thus limiting the amount of time during
which attacks can be successful.

5.2. Recommended Cipher Suites

Given the foregoing considerations, implementation of the following
cipher suites is RECOMMENDED:

o TLS_DHE_RSA_WITH_AES_128_GCM_SHA256

o TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256

o TLS_DHE_RSA_WITH_AES_256_GCM_SHA384

o TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384

We suggest that TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 be preferred in
general. See [RFC5289] for additional implementation details.

It is noted that those cipher suites are supported only in TLS 1.2
since they are authenticated encryption (AEAD) algorithms [RFC5116].

[RFC4492] allows clients and servers to negotiate ECDH parameters
(curves). Both clients and servers SHOULD include the "Supported
Elliptic Curves" extension [RFC4492]. For interoperability, clients
and servers SHOULD support the NIST P-256 (secp256r1) curve
[RFC4492]. In addition, clients SHOULD send an ec_point_formats
extension with a single element, "uncompressed".

5.3. Cipher Suite Negotiation Details

Clients SHOULD include TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 as the
first proposal to any server, unless they have prior knowledge that
the server cannot respond to a TLS 1.2 client_hello message.

Servers SHOULD prefer this cipher suite whenever it is proposed, even
if it is not the first proposal.

Clients are of course free to offer stronger cipher suites, e.g.
using AES-256; when they do, the server SHOULD prefer the stronger
cipher suite unless there are compelling reasons (e.g., seriously
degraded performance) to choose otherwise.

Note that other profiles of TLS 1.2 exist that use different cipher
suites. For example, [RFC6460] defines a profile that uses the
TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 and
TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 cipher suites.

This document is not an application profile standard, in the sense of
Sec. 9 of [RFC5246]. As a result, clients and servers are still
REQUIRED to support the mandatory TLS cipher suite,
TLS_RSA_WITH_AES_128_CBC_SHA.

5.4. Public Key Length

With a key exchange based on modular Diffie-Hellman ("DHE" cipher
suites), key lengths of at least 2048 bits are RECOMMENDED.

Rationale: because Diffie-Hellman keys of 1024 bits are estimated to
be roughly equivalent to 80-bit symmetric keys, it is better to use
longer keys for the "DHE" family of cipher suites. Unfortunately,
some existing software cannot handle (or cannot easily handle) key
lengths greater than 1024 bits. The most common workaround for these
systems is to prefer the "ECDHE" family of cipher suites instead of
the "DHE" family. For modular groups, key lengths of at least 2048
bits are estimated to be roughly equivalent to 112-bit symmetric keys
and might be sufficient for at least the next 10 years.

Servers SHOULD authenticate using 2048-bit certificates. In
addition, the use of SHA-256 fingerprints is RECOMMENDED (see
[CAB-Baseline] for more details). Clients SHOULD indicate to servers
that they request SHA-256, by using the "Signature Algorithms"
extension defined in TLS 1.2.

5.5. Modular vs. Elliptic Curve DH Cipher Suites

Not all TLS implementations support both modular and EC Diffie-
Hellman groups, as required by Section 5.2. Some implementations are
severely limited in the length of DH values. When such
implementations need to be accommodated, we recommend using (in
priority order):

1. Elliptic Curve DHE with negotiated parameters [RFC5289]

2. TLS_DHE_RSA_WITH_AES_128_GCM_SHA256 [RFC5288], with 2048-bit
    Diffie-Hellman parameters

3. The same cipher suite, with 1024-bit parameters.

Rationale: Elliptic Curve Cryptography is not universally deployed
for several reasons, including its complexity compared to modular
arithmetic and longstanding IPR concerns. On the other hand, there
are two related issues hindering effective use of modular Diffie-
Hellman cipher suites in TLS:

o There are no protocol mechanisms to negotiate the DH groups or
   parameter lengths supported by client and server.

o There are widely deployed client implementations that reject
   received DH parameters if they are longer than 1024 bits.

We note that with DHE and ECDHE cipher suites, the TLS master key
only depends on the Diffie Hellman parameters and not on the strength
of the RSA certificate; moreover, 1024 bit modular DH parameters are
generally considered insufficient at this time.

With modular ephemeral DH, deployers SHOULD carefully evaluate
interoperability vs. security considerations when configuring their
TLS endpoints.

5.6. Truncated HMAC

The truncated HMAC extension, defined in Sec. 7 of [RFC6066] does not
apply to the AEAD cipher suites recommended above. However it does
apply to most other TLS cipher suites. Its use has been shown to be
insecure in [PatersonRS11], and implementations MUST NOT use it.

6. IANA Considerations

This document requests no actions of IANA. [Note to RFC Editor:
please remove this whole section before publication.]

7. Security Considerations

   This entire document discusses the security practices directly
   affecting applications using the TLS protocol. This section contains
   broader security considerations related to technologies used in
   conjunction with or by TLS.

7.1. Host Name Validation

   Application authors should take note that TLS implementations
   frequently do not validate host names, and must therefore determine
   if the TLS implementation they are using does, and if not write their
   own validation code or consider changing the TLS implementation.

   It is noted that the requirements regarding host name validation (and
   in general, binding between the TLS layer and the protocol that runs
   above it) vary between different protocols. For HTTPS, these
   requirements are defined by Sec. 3 of [RFC2818].

   Readers are referred to [RFC6125] for further details regarding
   generic host name validation in the TLS context. In addition, the
   RFC contains a long list of example protocols, some of which
   implement a policy very different from HTTPS.

   If the host name is discovered indirectly and in an insecure manner
   (e.g., by an insecure DNS query for an MX or SRV record), it SHOULD
   NOT be used as a reference identifier [RFC6125] even when it matches
   the presented certificate. This proviso does not apply if the host
   name is discovered securely (for further discussion, see for example
   [I-D.ietf-dane-srv] and [I-D.ietf-dane-smtp]).

7.2. AES-GCM

   Sec. Section 5.2 above recommends the use of the AES-GCM
   authenticated encryption algorithm. Please refer to [RFC5246], Sec.
   11 for general security considerations when using TLS 1.2, and to
   [RFC5288], Sec. 6 for security considerations that apply specifically
   to AES-GCM when used with TLS.

7.3. Forward Secrecy

   Forward secrecy (also often called Perfect Forward Secrecy or "PFS",
   and defined in [RFC4949]) is a defense against an attacker who
   records encrypted conversations where the session keys are only
   encrypted with the communicating parties' long-term keys. Should the
   attacker be able to obtain these long-term keys at some point later
   in time, he will be able to decrypt the session keys and thus the
   entire conversation. In the context of TLS and DTLS, such compromise

of long-term keys is not entirely implausible. It can happen, for
example, due to:

o A client or server being attacked by some other attack vector, and
   the private key retrieved.

o A long-term key retrieved from a device that has been sold or
   otherwise decommissioned without prior wiping.

o A long-term key used on a device as a default key [Heninger2012].

o A key generated by a Trusted Third Party like a CA, and later
   retrieved from it either by extortion or compromise
   [Soghoian2011].

o A cryptographic break-through, or the use of asymmetric keys with
   insufficient length [Kleinjung2010].

PFS ensures in such cases that the session keys cannot be determined
even by an attacker who obtains the long-term keys some time after
the conversation. It also protects against an attacker who is in
possession of the long-term keys, but remains passive during the
conversation.

PFS is generally achieved by using the Diffie-Hellman scheme to
derive session keys. The Diffie-Hellman scheme has both parties
maintain private secrets and send parameters over the network as
modular powers over certain cyclic groups. The properties of the so-
called Discrete Logarithm Problem (DLP) allow to derive the session
keys without an eavesdropper being able to do so. There is currently
no known attack against DLP if sufficiently large parameters are
chosen. A variant of the Diffie-Hellman scheme uses Elliptic Curves
instead of the originally proposed modular arithmetics.

Unfortunately, many TLS/DTLS cipher suites were defined that do not
feature PFS, e.g. TLS_RSA_WITH_AES_256_CBC_SHA256. We thus advocate
strict use of PFS-only ciphers.

7.4. Diffie Hellman Exponent Reuse

For performance reasons, many TLS implementations reuse Diffie-
Hellman and Elliptic Curve Diffie-Hellman exponents across multiple
connections. Such reuse can result in major security issues:

o If exponents are reused for a long time (e.g., more than a few
   hours), an attacker who gains access to the host can decrypt
   previous connections. In other words, exponent reuse negates the
   effects of forward secrecy.

o TLS implementations that reuse exponents should test the DH public
   key they receive, in order to avoid some known attacks. These
   tests are not standardized in TLS at the time of writing. See
   [RFC6989] for recipient tests required of IKEv2 implementations
   that reuse DH exponents.

7.5. Certificate Revocation

Unfortunately there is currently no effective, Internet-scale
mechanism to affect certificate revocation:

o Certificate Revocation Lists (CRLs) are non-scalable and therefore
   rarely used.

o The On-Line Certification Status Protocol (OCSP) presents both
   scaling and privacy issues when used for heavy traffic Web
   servers. In addition, clients typically "soft-fail", meaning they
   do not abort the TLS connection if the OCSP server does not
   respond.

o OCSP stapling (Sec. 8 of [RFC6066]) resolves the operational
   issues with OCSP, but is still ineffective in the presence of a
   MITM attacker because they can simply ignore the client's request
   for a stapled OCSP response.

o OCSP stapling as defined in [RFC6066] does not extend to
   intermediate certificates used in a certificate chain. [RFC6961]
   addresses this shortcoming, but is a recent addition without much
   deployment.

o Proprietary mechanisms that embed revocation lists in the Web
   browser's configuration database cannot scale beyond a small
   number of the most heavily used Web servers.

The current consensus appears to be that OCSP stapling, combined with
a "must staple" mechanism similar to HSTS, would finally resolve this
problem; in particular when used together with the extension defined
in [RFC6961]. But such a mechanism has not been standardized yet.

8. Acknowledgments

We would like to thank Uri Blumenthal, Viktor Dukhovni, Stephen
Farrell, Simon Josefsson, Watson Ladd, Orit Levin, Johannes Merkle,
Bodo Moeller, Yoav Nir, Kenny Paterson, Patrick Pelletier, Tom
Ritter, Rich Salz, Aaron Zauner for their review and improvements.
Thanks to Brian Smith whose "browser cipher suites" page is a great
resource. Finally, thanks to all others who commented on the TLS,
UTA and other lists and are not mentioned here by name.

9. References

9.1. Normative References

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

[RFC2818]   Rescorla, E., "HTTP Over TLS", RFC 2818, May 2000.

[RFC4492]   Blake-Wilson, S., Bolyard, N., Gupta, V., Hawk, C., and B.
            Moeller, "Elliptic Curve Cryptography (ECC) Cipher Suites
            for Transport Layer Security (TLS)", RFC 4492, May 2006.

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

[RFC5288]   Salowey, J., Choudhury, A., and D. McGrew, "AES Galois
            Counter Mode (GCM) Cipher Suites for TLS", RFC 5288,
            August 2008.

[RFC5289]   Rescorla, E., "TLS Elliptic Curve Cipher Suites with SHA-
            256/384 and AES Galois Counter Mode (GCM)", RFC 5289,
            August 2008.

[RFC5746]   Rescorla, E., Ray, M., Dispensa, S., and N. Oskov,
            "Transport Layer Security (TLS) Renegotiation Indication
            Extension", RFC 5746, February 2010.

[RFC6125]   Saint-Andre, P. and J. Hodges, "Representation and
            Verification of Domain-Based Application Service Identity
            within Internet Public Key Infrastructure Using X.509
            (PKIX) Certificates in the Context of Transport Layer
            Security (TLS)", RFC 6125, March 2011.

[RFC6176]   Turner, S. and T. Polk, "Prohibiting Secure Sockets Layer
            (SSL) Version 2.0", RFC 6176, March 2011.

[RFC6347]   Rescorla, E. and N. Modadugu, "Datagram Transport Layer
            Security Version 1.2", RFC 6347, January 2012.

9.2. Informative References

[CAB-Baseline]
            CA/Browser Forum, , "Baseline Requirements for the
            Issuance and Management of Publicly-Trusted Certificates
            Version 1.1.6", 2013, <https://www.cabforum.org/
            documents.html>.

[DegabrieleP07]
          Degabriele, J. and K. Paterson, "Attacking the IPsec
          standards in encryption-only configurations", 2007,
          <http://dx.doi.org/10.1109/SP.2007.8>.

[Heninger2012]
          Heninger, N., Durumeric, Z., Wustrow, E., and J.
          Halderman, "Mining Your Ps and Qs: Detection of Widespread
          Weak Keys in Network Devices", Usenix Security Symposium
          2012, 2012.

[I-D.ietf-dane-smtp]
          Finch, T., "Secure SMTP using DNS-Based Authentication of
          Named Entities (DANE) TLSA records.", draft-ietf-dane-
          smtp-01 (work in progress), February 2013.

[I-D.ietf-dane-srv]
          Finch, T., Miller, M., and P. Saint-Andre, "Using DNS-
          Based Authentication of Named Entities (DANE) TLSA Records
          with SRV Records", draft-ietf-dane-srv-07 (work in
          progress), July 2014.

[I-D.ietf-tls-prohibiting-rc4]
          Popov, A., "Prohibiting RC4 Cipher Suites", draft-ietf-
          tls-prohibiting-rc4-00 (work in progress), July 2014.

[I-D.ietf-uta-tls-attacks]
          Sheffer, Y., Holz, R., and P. Saint-Andre, "Summarizing
          Current Attacks on TLS and DTLS", draft-ietf-uta-tls-
          attacks-04 (work in progress), September 2014.

[Kleinjung2010]
          Kleinjung, T., "Factorization of a 768-Bit RSA Modulus",
          CRYPTO 10, 2010, <https://eprint.iacr.org/2010/006.pdf>.

[PatersonRS11]
          Paterson, K., Ristenpart, T., and T. Shrimpton, "Tag size
          does matter: attacks and proofs for the TLS record
          protocol", 2011,
          <http://dx.doi.org/10.1007/978-3-642-25385-0_20>.

[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.

   [RFC4347] Rescorla, E. and N. Modadugu, "Datagram Transport Layer
              Security", RFC 4347, April 2006.

   [RFC4949] Shirey, R., "Internet Security Glossary, Version 2", RFC
              4949, August 2007.

   [RFC5077] Salowey, J., Zhou, H., Eronen, P., and H. Tschofenig,
              "Transport Layer Security (TLS) Session Resumption without
              Server-Side State", RFC 5077, January 2008.

   [RFC5116] McGrew, D., "An Interface and Algorithms for Authenticated
              Encryption", RFC 5116, January 2008.

   [RFC6066] Eastlake, D., "Transport Layer Security (TLS) Extensions:
              Extension Definitions", RFC 6066, January 2011.

   [RFC6101] Freier, A., Karlton, P., and P. Kocher, "The Secure
              Sockets Layer (SSL) Protocol Version 3.0", RFC 6101,
              August 2011.

   [RFC6460] Salter, M. and R. Housley, "Suite B Profile for Transport
              Layer Security (TLS)", RFC 6460, January 2012.

   [RFC6797] Hodges, J., Jackson, C., and A. Barth, "HTTP Strict
              Transport Security (HSTS)", RFC 6797, November 2012.

   [RFC6961] Pettersen, Y., "The Transport Layer Security (TLS)
              Multiple Certificate Status Request Extension", RFC 6961,
              June 2013.

   [RFC6989] Sheffer, Y. and S. Fluhrer, "Additional Diffie-Hellman
              Tests for the Internet Key Exchange Protocol Version 2
              (IKEv2)", RFC 6989, July 2013.

   [Soghoian2011]
              Soghoian, C. and S. Stamm, "Certified lies: Detecting and
              defeating government interception attacks against SSL.",
              Proc. 15th Int. Conf. Financial Cryptography and Data
              Security , 2011.

   [triple-handshake]
              Delignat-Lavaud, A., Bhargavan, K., and A. Pironti,
              "Triple Handshakes Considered Harmful: Breaking and Fixing
              Authentication over TLS", 2014, <https://secure-
              resumption.com/>.

Appendix A. **Change Log**

   Note to RFC Editor: please remove this section before publication.

A.1.   **draft-ietf-uta-tls-bcp-04**

   o Some cleanup, and input from TLS WG discussion on applicability.

A.2.   **draft-ietf-uta-tls-bcp-03**

   o Disallow truncated HMAC.

   o Applicability to DTLS.

   o Some more text restructuring.

   o Host name validation is sometimes irrelevant.

   o HSTS: MUST implement, SHOULD deploy.

   o Session identities are not protected, only tickets are.

   o Clarified the target audience.

A.3.   **draft-ietf-uta-tls-bcp-02**

   o Rearranged some sections for clarity and re-styled the text so
      that normative text is followed by rationale where possible.

   o Removed the recommendation to use Brainpool curves.

   o Triple Handshake mitigation.

   o MUST NOT negotiate algorithms lower than 112 bits of security.

   o MUST implement SNI, but use per local policy.

   o Changed SHOULD NOT negotiate or fall back to SSLv3 to MUST NOT.

   o Added hostname validation.

   o Non-normative discussion of DH exponent reuse.

A.4.   **draft-ietf-tls-bcp-01**

   o Clarified that specific TLS-using protocols may have stricter
      requirements.

o Changed TLS 1.0 from MAY to SHOULD NOT.

o Added discussion of "optional TLS" and HSTS.

o Recommended use of the Signature Algorithm and Renegotiation Info
   extensions.

o Use of a strong cipher for a resumption ticket: changed SHOULD to
   MUST.

o Added an informational discussion of certificate revocation, but
   no recommendations.

A.5. [draft-ietf-tls-bcp-00](#)

o Initial WG version, with only updated references.

A.6. [draft-sheffer-tls-bcp-02](#)

o Reorganized the content to focus on recommendations.

o Moved description of attacks to a separate document ([draft-sheffer-uta-tls-attacks](#)).

o Strengthened recommendations regarding session resumption.

A.7. [draft-sheffer-tls-bcp-01](#)

o Clarified our motivation in the introduction.

o Added a section justifying the need for PFS.

o Added recommendations for RSA and DH parameter lengths. Moved
   from DHE to ECDHE, with a discussion on whether/when DHE is
   appropriate.

o Recommendation to avoid fallback to SSLv3.

o Initial information about browser support - more still needed!

o More clarity on compression.

o Client can offer stronger cipher suites.

o Discussion of the regular TLS mandatory cipher suite.

A.8. draft-sheffer-tls-bcp-00

   o Initial version.

Authors' Addresses

   Yaron Sheffer
   Porticor
   29 HaHarash St.
   Hod HaSharon 4501303
   Israel

   Email: yaronf.ietf@gmail.com

   Ralph Holz
   Technische Universitaet Muenchen
   Boltzmannstr. 3
   Garching 85748
   Germany

   Email: holz@net.in.tum.de

   Peter Saint-Andre
   &yet
   P.O. Box 787
   Parker, CO 80134
   USA

   Email: peter@andyet.com