        Problems with the Public Key Infrastructure (PKI) for the World Wide Web
                    draft-housley-web-pki-problems-01.txt

Abstract

   This document describes the technical and non-technical problems with
   the current Public Key Infrastructure (PKI) used for the World Wide
   Web. Some potential technical improvements are considered, and some
   non-technical approaches to improvements 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 April 3, 2016.

Table of Contents

1. Introduction

   There are many technical and non-technical problems with the current
   Public Key Infrastructure (PKI) used for the World Wide Web. This
   document describes these problems, considers some potential technical
   improvements, and discusses some non-technical approaches to
   improvements.

   The Web PKI makes use of certificates as described in RFC 5280
   [RFC5280]. These certificates are primarily used with Transport
   Layer Security (TLS) RFC 5246 [RFC5246].

2. Very Brief Description of the Web PKI

   These topics are to be covered in this section:

   o Certificate bind a name or names to a public key

o Enrollment process makes sure:

   * it is the right name

   * confirms that party actually holds the private key

o Explain the entities

3. Technical Improvements to the Web PKI

Over the years, many technical improvements have been made to the Web
PKI. This section discusses sever problems and the technical
problems that have been made to address them. This history sets the
stage for suggestions for additional improvements in other sections
of this document.

3.1. Weak Cryptographic Algorithms or Short Public Keys

Over the years, the digital signature algorithms, one-way hash
functions, and public key sizes that are considered strong have
changed. This is not a surprise. Cryptographic algorithms age; they
become weaker with time. As new cryptanalysis techniques are
developed and computing capabilities improve, the work factor to
break a particular cryptographic algorithm will reduce. For this
reason, the algorithms and key sizes used in the Web PKI need to
migrate over time. A reasonable choice of algorithm or key size
needs to be evaluated periodically, and a transition may be needed
before the expected lifetime expires.

The browser vendors have been trying to manage algorithm and key size
transitions, but a long-lived trust anchor or intermediate CA
certificate can have a huge number of subordinate certificates. So,
removing a one because it uses a weak cryptographic algorithm or a
short public key can have a significant impact.

As a result, some valid trust anchors and certificates contain
cryptographic algorithms after weakness has been discovered and
widely known. Similarly, valid trust anchors and certificates
contain public keys after computational resources available to
attackers have rendered them too weak. We have seen a very
successful migration away from certificates that use the MD2 or MD5
one-way hash functions. However, there are still a great number of
certificates that use SHA-1 and 1024-bit RSA public keys, and these
should be replaced.

Today, the algorithms and key sizes used by a CA to sign certificates
with a traditional lifespan should offer at least 128 bits of
security. SHA-256 is a widely studied one-way hash function that

meets this requirement. RSA with a public key of at least 2048 bits
or ECDSA with a public key of at least 256 bits are widely studied
digital signature algorithms that meet this requirement.

3.2. Certificate Status Checking

Several years ago, many browsers do not perform certificate status
checks by default. That is, browsers did not check whether the
issuing CA has revoked the certificate unless the user explicitly
adjusted a setting to enable this feature. This check can be made by
fetching the most recent certificate revocation list (CRL) RFC 5280
[RFC5280], or this check can use the Online Certificate Status
Protocol (OCSP) RFC 6960 [RFC6960]. The location of the CRL or the
OCSP responder is usually found in the certificate itself. Either
one of these approaches add latency. The desire to provide a snappy
user experience is a significant reason that this feature was not
turned on by default.

Certificate status checking needs to be used at all times. Several
techniques have been tried by CAs and browsers to make certificate
status checking more efficient. Many CAs are using of Content
Delivery Networks (CDNs) by CAs to deliver CRLs and OCSP responses,
resulting in very high availability and low latency. Yet, browser
vendors are still reluctant to perform standard-based status checking
by default for every session.

3.2.1. Short-lived Certificates

Short-lived certificates are an excellent way to reduce the need for
certificate status checking. The shorter the life of the
certificate, the less time there is for anything to go wrong. If the
lifetime is short enough, policy might allow certificate status
checking can be skipped altogether. In practice, implementation of
short-lived certificates requires automation to assist web server
administrators, which is a topic that is discussed elsewhere in this
document.

3.2.2. CRL Distribution Points

The certificate revocation list distribution point (CRLDP)
certificate extension RFC 5280 [RFC5280] allows a CA to control the
maximum size of the CRLs that they issue. This helps in two ways.
First, the amount of storage needed by the browser to cache CRLs is
reduced. Second, and more importantly, the amount of time it takes
to download a CRL can be scoped, so that the amount of time needed to
fetch any single CRL is reasonable.

Few CAs take advantage of the CRLDP certificate extension to limit
the size of CRLs. In fact, there are several CAs that publish
extremely large CRLs. Browsers never want to suffer the latency
associated with large CRLs, and some ignore the CRLDP extension when
it is present. Browsers tend to avoid the use of CRLs altogether.

3.2.3. Proprietary Revocation Checks

Some browser vendors provide a proprietary mechanism for revocation
checking. These mechanisms obtain revocation status information once
per day for the entire Web PKI in a very compact form. No network
traffic is generated at the time that a certificate is being
validated, so there is no latency associated with revocation status
checking. The browser vendor infrastructure performs daily checks of
the Web PKI, and then the results are assembled in a proprietary
format and made available to the browser. These checks only cover
the trust anchor store for that browser vendor, so any trust anchors
added by the user cannot be checked in this manner.

3.2.4. OCSP Stapling

Browsers can avoid transmission of CRLs altogether by using the
Online Certificate Status Protocol (OCSP) RFC 6960 [RFC6960] to check
the validity of web server certificates. The TLS Certificate Status
Request extension is defined in Section 8 of RFC 6066 [RFC6066]. In
addition, RFC 6961 [RFC6961] defines the TLS Multiple Certificate
Status Request extension, which allows the web server to provide
status information about its own certificate and also the status of
intermediate certificates in the certification path. By including
this extension in the TLS handshake, the browser asks the web server
to provide an OCSP response in addition to its certificate. This
approach greatly reduces the number of round trips by the browser to
check the status of each certificate in the path. In addition, the
web server can cache the OCSP response for a period of time, avoiding
additional latency. Even in the cases where the web server needs to
contact the OCSP responder, the web server usually has a higher
bandwidth connection than the browser to do so.

The provision of the time-stamped OCSP response in the TLS handshake
is referred to as "stapling" the OCSP response to the TLS handshake.
If the browser does not receive a stapled OCSP response, it can
contact the OCSP responder directly. In addition, the MUST_STAPLE
feature [TLSFEATURE] can be used to insist that OCSP stapling be
used.

When every browser that connects to a high volume website performs
its own OCSP lookup, the OCSP responder must handle a real-time
response to every browser. OCSP stapling can avoid enormous volumes

   of OCSP requests for certificates of popular websites, so stapling
   can significantly reduce the cost of providing an OCSP service.

   OCSP stapling can also improve user privacy, since the web server,
   not the browser, contacts the OCSP responder. In this way, the OCSP
   responder is not able to determine which browsers are checking the
   validity of certificate for websites.

   Many web site are taking advantage of OCSP sampling. At the time of
   this writing, browser venders report that about 12% the the
   transactions use OCSP sampling, and the number is on the rise.

3.3. Surprising Certificates

   All of the CAs in the trust store are equally trusted for the entire
   domain name space, so any CA can issue for any domain name. In fact,
   there have been certificates issued by CAs that are surprising to the
   legitimate owner of a domain. The domain name owner is surprised
   because they did not request the certificates. They are initially
   unaware that a CA has issued a certificate that contains their domain
   name, and once the surprising certificate is discovered, it can be
   very difficult for the legitimate domain name owner to get it
   revoked. Further, browsers and other relying parties cannot
   distinguish a certificate that the legitimate domain name owner
   requested from an surprising one.

   Since all of the CAs in the trust store are equally trusted, any CA
   can issue a certificate for any domain name. There are known cases
   where attackers have thwarted the CA protections and issued
   certificates that were then used to mount attacks against the users
   of that web site [FOXIT]. For this reason, all of the CAs listed in
   the trust store must be very well protected.

   The Baseline Requirements produced by the CA/Browser Forum [CAB2014]
   tell CAs the checks that need to be performed before a certificate is
   issued. In addition, WebTrust [WEBTRUST] provides the audit
   requirements for CAs, and browser vendors will remove a CA from the
   trust anchor store if successful audit reports are not made
   available.

   When a CA issues a certificate to a subordinate CA, the inclusion of
   widely supported certificate extensions can reduce set of privileges
   given to the sub-CA. This aligns with the traditional security
   practice of least privilege, where only the privileges needed to
   perform the envisioned tasks are provided. However, many sub-CAs
   have certificates that pass along the full powers of the CA, creating
   additional high-pay-off targets for attackers, and these sub-CAs may

not be held to the same certificate issuance requirements and audit
requirement as the parent CA.

Some major implementations have not fully implemented the mechanisms
necessary to reduce sub-CA privileges. For example, RFC 5280
[RFC5280] includes the specification of name constraints, and the CA/
Browser Forum guidelines [CAB2014] encourage the use dNSNames in
permittedSubtrees within the name Constraints extension. Despite
this situation, one major browser does not support name constraints,
and as a result, CAs are reluctant to use them. Further, global CAs
are prepared to issue certificates within every top-level domain,
including ones that are newly-approved. It is not practical for
these global CAs to use name constraints in their sub-CA
certificates.

As a result of procedural failures or attacks, surprising
certificates are being issued. Several mechanisms have been defined
to avoid the issuance of surprising certificates or prevent browsers
from accepting them.

3.3.1. Certificate Authority Authorization (CAA)

The Certificate Authority Authorization (CAA) [RFC6844] DNS resource
record allows a domain administrator to specify one or more CA that
is authorized to issue certificates that include the domain name.
Then, a trustworthy CA will refuse to issue a certificate for a
domain name that has a CAA resource record that does not explicitly
name the CA.

To date, only one major CA performs this check, and there is no
indication that other CAs are planning to add this check in the near
future.

3.3.2. HTTP Public Key Pinning (HPKP)

HTTP Public Key Pinning (HPKP) [RFC7469] allows a web server to
instruct browsers to remember the server's public key fingerprints
for a period of time. The fingerprint is a one-way hash of the
subject public key information in the certificate. The Public-Key-
Pins header provides a maximum retention period, fingerprints of the
web server certificate, and optionally fingerprints for backup
certificates. The act of saving of the fingerprints is referred to
as "pinning". During pin lifetime, browsers require that the same
web server present a certificate chain that includes a public key
that matches one of the retained fingerprints. This prevents
impersonation of the website with a surprising certificate.

A website can choose to pin the CA certificate so that the browser
will accept only valid certificates for the website domain that are
issued by that CA. Alternatively, the website can choose to pin
their own certificate and at least one backup certificate in case the
current certificate needs to be replaced due to a compromised server.

Some browser vendors also pin certificates by hardcoding fingerprints
of very well known websites.

When HPKP is used, browsers may be able to detect a man-in-the-
middle. Sometimes the man-in-the-middle is an attacker, and other
times a service provider purposefully terminates the TLS at a
location other than the web server. One example became very public
in February 2012 when Trustwave admitted that it had issued a
subordinate CA certificate for use by a company to inspect corporate
network traffic [LC2012]. When HPKP is used, the browser user will
be notified if the key-pining is violated, unless the violating
certificate can be validated to a locally installed trust anchor. In
this situation, the browser is assuming that the user intended to
explicitly trust the certificate.

3.3.3. HTTP Strict Transport Security (HSTS)

HTTP Strict Transport Security (HSTS) [RFC6797] is a security policy
mechanism that protects secure websites against downgrade attacks,
and it greatly simplifies protection against cookie hijacking. The
presence of the Strict-Transport-Security header tells browsers that
all interactions with this web server should never use HTTP without
TLS, providing protection against eavesdropping and active network
attacks.

When a web server includes the Strict-Transport-Security header, the
browser is expected to do two things. First, the browser
automatically turns any insecure links into secure ones. For
instance, "http://mysite.example.com/mypage/" will be changed to
"https://mysite.example.com/mypage/". Second, if the TLS Handshake
results in some failure, such as the certificate cannot be validated,
then an error message is displayed and the user is denied access the
web application.

3.3.4. DNS-Based Authentication of Named Entities (DANE)

The DNS-Based Authentication of Named Entities (DANE) [RFC6698]
allows domain administrators to specify the raw public keys or
certificates that are used by web servers in their domain. DANE
leverages the DNS Security Extensions (DNSSEC) [RFC4034][RFC4035],
which provides digital signatures over DNS zones that are validated
with keys that are bound to the domain name of the signed zone. The

keys associated with a domain name can only be signed by a key
associated with the parent of that domain name. For example, the
DNSSEC keys for "www.example.com" can only be signed by the DNSSEC
keys for "example.com". Therefore, a malicious actor can only
compromise the keys of their own subdomains. Like the Web PKI,
DNSSEC relies on public keys used to validate chains of signatures,
but DNSSEC has a single root domain as opposed to a multiplicity of
trusted CAs.

DANE binds raw public keys or certificates to DNS names. The domain
administrator is the one that vouches for the binding of the public
key or the certificate to the domain name by adding the TSLA records
to the zone and then signing the zone. In this way, the same
administrator is responsible for managing the DNS names themselves
and associated public keys or certificates with those names. DANE
restricts the scope of assertions that can be made, forcing them to
be consistent with the DNS naming hierarchy.

In addition, DNSSEC reduces opportunities for redirection attacks by
binding the domain name to the public key or certificate.

Some Web PKI certificates are being posted in TLSA records, but
browsers expect to receive the the server certificate in the TLS
handshake, and there is little incentive to confirm that the received
certificate matches the one posted in the DNS. For this reason, work
has begun on a TLS extension that will allow the DNSSEC-protected
information to be provided in the handshake, which will eliminate the
latency [TLSCHAIN].

3.3.5. Certificate Transparency

Certificate Transparency (CT) [RFC6962] offers a mechanism to detect
mis-issued certificates, and once detected, administrators and CAs
can take the necessary actions to revoke the mis-issued certificates.

When requesting a certificate, the administrator can request the CA
to include an embedded Signed Certificate Timestamp (SCT) in the
certificate to ensure that their legitimate certificate is logged
with one or more CT log.

An administrator, or another party acting on behalf of the
administrator, is able to monitor one or more CT log to which a pre-
certificate or certificate is submitted, and detect the logging of a
pre-certificate or certificate that contains their domain name. When
such a pre-certificate or certificate is detected, the CA can be
contacted to to get the mis-issued certificate revoked.

In the future, a browser may choose to reject certificates that do
not contain an SCT, and potentially notify the website administrator
or CA when they encounter such a certificate. Such reporting will
help detect mis-issuance of certificates and lead to their
revocation.

3.4. Automation for Server Administrators

There have been several attempts to provide automation for routine
tasks that are performed by web server administrators, such as
certificate renewal. For example, some commercial tools offer
automated certificate renewal and installation [DCEI][SSLM]. Also,
at least one proposal was brought to the IETF that allows a web
server automate obtaining and renewing certificates [PHBOB]. Without
automation, there are many manual steps involved in getting a
certificate from a CA, and to date none of these attempts at
automation have not enjoyed widespread interoperability and adoption.
There are at least two ways that this impacts web security. First,
many web sites do not have a certificate at all. The cost, time, and
effort are too great for the system administrator to go through the
effort, especially if the web site does not offer anything for
purchase. Second, once a certificate is obtained, a replacement is
not obtained until the current one expires. Automation can reduce
the amount of time that an administrator needs to dedicate to
certificate management, and it can make certificate renewal timely
and automatic.

The IETF ACME working group [ACMEWG] is working on protocols that
will provide system administrators an automated way to enroll and
renew their certificates. The expectation is that these
specifications will lead to widely available and interoperable tools
for system administrators. The expectation is that these protocols
and tools will be supported by all web server environments and CAs,
which will greatly reduce complexity and cost.

4. Policy and Process Improvements to the Web PKI

As with many technologies, the issues and complexities associated
with Web PKI use and deployment are just as much policy and process
as technical. These have evolved over time as well. This section
discusses the ways that business models and operational policies and
processes impact the Web PKI.

4.1. Determination of the Trusted Certificate Authorities

A very basic question for users of the Web PKI is "Who do you trust?"
The system for determining which CAs are added to or removed from the
trust store in browsers has been perceived by some as opaque and

confusing. As mentioned earlier, the CA/Browser Forum has developed
baseline requirements for the management and issuance of certificates
[CAB2014] for individual CAs. However, the process by which an
individual CA gets added to the trust store for each of the major
browsers is not straightforward. The individual browser vendors
determine what should and should not be trusted by including those
trusted CAs in their trust store. They do this by leveraging the
AICPA/CICA WebTrust Program for Certification Authorities [WEBTRUST].
This program provides auditing requirements and a trust mark for CAs.
Failure to pass an audit can result in the CA being removed from the
trust store.

Once the browser has shipped, how does a user know which CAs are
trusted or what has changed recently. For an informed user,
information about which CAs have been added to or deleted from the
browser trust store can be found in the release notes. Users can
also examine the policies of the various CAs which would have been
developed and posted for the WebTrust Program. However, this may be
considered a fairly high barrier for the average user. There are
also options to make local modifications by educated users, but there
is little understanding about the implications of these choices. How
does an individual, organization, or enterprise really determine if a
particular CA is trustworthy? Do the default choices inherited from
the browser vendors truly represent the organization's trust model?
What constitutes sufficiently bad behavior by a CA to cause removal
from the trust store?

One form of bad behavior by CAs is the mis-issuance of certificates.
This mis-issuance can be either an honest mistake by the CA,
malicious behavior by the CA, or a case where an external party has
duped the CA into the mis-issuance. When a CA has delegated
authority to a sub-CA, and then the sub-CA issued bad certificates
either unintentionally or maliciously, the CA is able to deny
responsibility for the actions of the sub-CA. However, the CA may be
the only party that can revoke the sub-CA certificate to protect the
overall Web PKI.

Another complication with CAs and the trust store maintained by the
browser vendor is an enterprise managed PKI. For example, the US
Department of Defense operates its own PKI. In this case, the
enterprise maintains its own PKI for the exclusive use of the
enterprise itself. A bridge CA may be used to connect related
enterprises. The complication in this approach is that the
revocation mechanisms don't work with any additions that have been
made by the enterprise. See Section 3.2.3 on proprietary revocation
checks.

What constitutes sufficiently bad behavior by a CA to cause removal
from the trust store? The guidelines provided by the WebTrust
program [WEBTRUST] provide a framework, but the implications of
removing a CA can be significant. There may be a few very large CAs
that are critical to significant portions of Internet infrastructure.
Removing one of these trusted CAs can have a significant impact on a
large cross section of Internet users.

4.2. Governance Structures for the Web PKI

There are a number of organizations that play significant roles in
the operation of the Web PKI, including the CAB Forum, the WebTrust
Program, and the browser vendors. These organizations act on behalf
of the entire Internet community. Transparency in these operations
is vital to basic trust in the Web PKI. As one example, in the past
the CAB Forum was perceived as being a closed forum; however, some
changes were made to the operational procedures to allow more
visibility if not actual participation in the process [CAB1.2]. How
do we ensure that these processes continue to evolve in an open,
inclusive, and transparent manner? Currently, as the name implies,
the CAB Forum members represent CAs and browser vendors. How do we
ensure that relying parties a voice in this forum?

Since the Web PKI is widespread, applications beyond the World Wide
Web are making use of the Web PKI. For example, the Web PKI is used
to secure the connections between SMTP servers. In these
environments, the browser-centric capabilities are unavailable. For
example, see Section 3.2.3 on proprietary revocation checks. The
current governance structure does not provide a way for these other
applications to participate. How do we ensure that these other
applications get a voice in this forum?

5. Other Considerations for Improving the Web PKI

Other factors impact the usability and reliability of the Web PKI.
One factor is time synchronization. As time synchronization
infrastructure is made more secure, this infrastrucre will require
the use of certificates to authenticate time servers. However,
certificate infrastructure is reliant on quality time synchronization
as well, creating a boot strapping issue.

6. Security Considerations

Many people find browser error messages related to certificates
confusing. Good man-machine interfaces are always difficult, but in
this situation users are unable to understand the risks that they are
accepting by clicking "okay". This aspect of browser usability needs
to be improved for users to make better security choices.

7. IANA Considerations

   None.

   {{{ RFC Editor: Please remove this section prior to publication. }}}

8. References

8.1. Normative References

   [RFC5280] Cooper, D., Santesson, S., Farrell, S., Boeyen, S.,
              Housley, R., and W. Polk, "Internet X.509 Public Key
              Infrastructure Certificate and Certificate Revocation List
              (CRL) Profile", RFC 5280, DOI 10.17487/RFC5280, May 2008,
              <http://www.rfc-editor.org/info/rfc5280>.

8.2. Informative References

   [ACMEWG]   IETF, "Charter for Automated Certificate Management
              Environment (acme) Working Group", June 2015,
              <https://datatracker.ietf.org/doc/charter-ietf-acme/>.

   [CAB1.2]   CA/Browser Forum, "Bylaws of the CA/Browser Forum",
              October 2014, <https://cabforum.org/wp-content/uploads/CA-
              Browser-Forum-Bylaws-v.1.2.pdf>.

   [CAB2014] CA/Browser Forum, "CA/Browser Forum Baseline Requirements
              for the Issuance and Management of Publicly-Trusted
              Certificates, v.1.2.2", October 2014,
              <https://cabforum.org/wp-content/uploads/BRv1.2.2.pdf>.

   [DCEI]   DigiCert Inc, "Express Install(TM): Automate SSL
              Certificate Installation and HTTPS Configuration", AUGUST
              2015, <https://www.digicert.com/express-install/>.

   [FOXIT]   Prins, J., "DigiNotar Certificate Authority breach:
              "Operation Black Tulip"", September 2011,
              <http://www.rijksoverheid.nl/bestanden/documenten-en-
              publicaties/rapporten/2011/09/05/
              diginotar-public-report-version-1/
              rapport-fox-it-operation-black-tulip-v1-0.pdf>.

   [LC2012]   Constantin, L., "Trustwave admits issuing man-in-the-
              middle digital certificate; Mozilla debates punishment",
              February 2012,
              <http://www.computerworld.com/article/2501291/internet/
              trustwave-admits-issuing-man-in-the-middle-digital-
              certificate--mozilla-debates-punishment.html>.

[LE]   Internet Security Research Group, "Let's Encrypt", August
            2015, <https://letsencrypt.org/>.

[PHBOB]   Hallam-Baker, P., "OmniBroker Publication Protocol",
            draft-hallambaker-omnipublish-00 (work in progress), May
            2014.

[RFC4034]   Arends, R., Austein, R., Larson, M., Massey, D., and S.
            Rose, "Resource Records for the DNS Security Extensions",
            RFC 4034, DOI 10.17487/RFC4034, March 2005,
            <http://www.rfc-editor.org/info/rfc4034>.

[RFC4035]   Arends, R., Austein, R., Larson, M., Massey, D., and S.
            Rose, "Protocol Modifications for the DNS Security
            Extensions", RFC 4035, DOI 10.17487/RFC4035, March 2005,
            <http://www.rfc-editor.org/info/rfc4035>.

[RFC5246]   Dierks, T. and E. Rescorla, "The Transport Layer Security
            (TLS) Protocol Version 1.2", RFC 5246, DOI 10.17487/
            RFC5246, August 2008,
            <http://www.rfc-editor.org/info/rfc5246>.

[RFC6066]   Eastlake 3rd, D., "Transport Layer Security (TLS)
            Extensions: Extension Definitions", RFC 6066, DOI
            10.17487/RFC6066, January 2011,
            <http://www.rfc-editor.org/info/rfc6066>.

[RFC6698]   Hoffman, P. and J. Schlyter, "The DNS-Based Authentication
            of Named Entities (DANE) Transport Layer Security (TLS)
            Protocol: TLSA", RFC 6698, DOI 10.17487/RFC6698, August
            2012, <http://www.rfc-editor.org/info/rfc6698>.

[RFC6797]   Hodges, J., Jackson, C., and A. Barth, "HTTP Strict
            Transport Security (HSTS)", RFC 6797, DOI 10.17487/
            RFC6797, November 2012,
            <http://www.rfc-editor.org/info/rfc6797>.

[RFC6844]   Hallam-Baker, P. and R. Stradling, "DNS Certification
            Authority Authorization (CAA) Resource Record", RFC 6844,
            DOI 10.17487/RFC6844, January 2013,
            <http://www.rfc-editor.org/info/rfc6844>.

[RFC6960]   Santesson, S., Myers, M., Ankney, R., Malpani, A.,
            Galperin, S., and C. Adams, "X.509 Internet Public Key
            Infrastructure Online Certificate Status Protocol - OCSP",
            RFC 6960, DOI 10.17487/RFC6960, June 2013,
            <http://www.rfc-editor.org/info/rfc6960>.

   [RFC6961] Pettersen, Y., "The Transport Layer Security (TLS)
              Multiple Certificate Status Request Extension", RFC 6961,
              DOI 10.17487/RFC6961, June 2013,
              <http://www.rfc-editor.org/info/rfc6961>.

   [RFC6962] Laurie, B., Langley, A., and E. Kasper, "Certificate
              Transparency", RFC 6962, DOI 10.17487/RFC6962, June 2013,
              <http://www.rfc-editor.org/info/rfc6962>.

   [RFC7469] Evans, C., Palmer, C., and R. Sleevi, "Public Key Pinning
              Extension for HTTP", RFC 7469, DOI 10.17487/RFC7469, April
              2015, <http://www.rfc-editor.org/info/rfc7469>.

   [SSLM]   Opsmate, Inc., "SSLMate: Secure your website the easy
              way", August 2015, <https://sslmate.com/>.

   [TLSCHAIN]
              Shore, M., Barnes, R., Huque, S., and W. Toorop, "X.509v3
              TLS Feature Extension", draft-shore-tls-dnssec-chain-
              extension-01 (work in progress), July 2015.

   [TLSFEATURE]
              Hallam-Baker, P., "X.509v3 TLS Feature Extension", draft-
              hallambaker-tlsfeature-10 (work in progress), July 2015.

   [WEBTRUST]
              CPA Canada, "WebTrust Program for Certification
              Authorities", August 2015, <http://www.webtrust.org/
              homepage-documents/item27839.aspx>.

Appendix A. Acknowledgements

   This document has been developed within the IAB Privacy and Security
   Program. The authors greatly appreciate the review and suggestions
   provided by Rick Andrews, Mary Barnes, Richard Barnes, Marc Blanchet,
   Alissa Cooper, Nick Doty, Stephen Farrell, Joe Hall, Ted Hardie,
   Ralph Holz, Christian Huitema, Eliot Lear, Xing Li, Lucy Lynch,
   Gervase Markham, Andrei Robachevsky, Thomas Roessler, Jeremy Rowley,
   Christine Runnegar, Jakob Schlyter, Wendy Seltzer, Brian Trammell,
   and Juan Carlos Zuniga.

Appendix B. IAB Members at the Time of Approval

   {{{ RFC Editor: Please add the names to the IAB members at the time
   that this document is put into the RFC Editor queue. }}}

Authors' Addresses

   Russ Housley
   Vigil Security
   918 Spring Knoll Drive
   Herndon, VA 20170
   USA

   Email: housley@vigilsec.com

   Karen O'Donoghue
   Internet Society
   1775 Wiehle Ave #201
   Reston, VA 20190
   USA

   Email: odonoghue@isoc.org