Internet Architecture Board
Internet-Draft
Intended status: Informational
Expires: January 7, 2016

R. Housley
Vigil Security
K. O'Donoghue
Internet Society
July 6, 2015

Problems with the Public Key Infrastructure (PKI) for the World Wide Web

draft-housley-web-pki-problems-00.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 January 7, 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:

- Certificate bind a name or names to a public key
- Enrollment process makes sure:
  - it is the right name
  - confirms that party actually holds the private key
- Explain the entities

3. Technical Problems with the Web PKI

There are many problems with the Web PKI, and this section discusses many of them.

3.1. No certificate status checking

Many browsers do not perform certificate status checks by default. That is they do not check whether the issuing CA has revoked the certificate unless the user explicitly adjusts a setting that turns on 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 the reason that this feature is not turned on by default.

3.2. Unexpected 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, the legitimate owner of the domain name might be unaware that a CA has issued a certificate that contains their domain name. Once the unexpected certificate is discovered, it can be very difficult to get it revoked. Further, browsers and other relying parties cannot distinguish a certificate that the legitimate domain name owner requested from an unexpected one.

3.3. All domain names are at risk from the failure of any CA

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 used to mount attacks against users of the web sites names in the certificate [FOXIT]. For this reason, all of the CAs listed in the trust store must be very well protected.

3.4. CAs issue 'full powers' to subordinate CAs

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, and additional high-pay-off targets for attackers are created.

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, at least one major browser does not support name constraints, and as a result, CAs are reluctant to use name constraints.

3.5. No automation for server operators

There are many manual steps involved in getting a certificate from a CA. There are at least two ways that this impacts web security. First, many 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 a system administrator needs to dedicate to certificate management, and it can make certificate renewal timely and automatic.

3.6. Attacks are not reported when they are detected

When browsers are able to detect a man-in-the-middle, they do not report this situation to the user. 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]. Regardless of the reason for the man-in-the-middle browser users should be aware of them every time that they are detected. If the man-in-the-middle is an attacker, then they are thwarted. If the man-in-the-middle is the service provider, then the browser user is aware that the traffic is not protected to the ultimate destination.

3.7. Weak cryptographic algorithms or short keys

Many certificates contain weak cryptographic algorithms or contain public keys that are too short. In many cases, this is because algorithm and key size choices that were reasonable when they were chosen and then put in a certificate with an extremely long lifetime. As a result, valid certificates contain cryptographic algorithms after weakness has been discovered and widely known. For example, certificates that use the MD2 or MD5 one-way hash functions should be replaced. Similarly, valid certificates contain public keys after computational resources available to attackers has rendered them too weak. For example, certificates that contain RSA public keys shorter than 1024 bits 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 2048 bits or ECDSA with a public key of 256 bits are widely studied digital signature algorithms that meet this requirement.

4. Non-technical Problems with the Web PKI

While there are several technical issues that impact the widespread usage of the Web PKI, there are also a number of non-technical or process problems that have impeded adoption and deployment. This section discusses the ways that business models and operational processes are hindering the Web PKI.

4.1. Determination of the Trusted Certificate Authorities

The current system for determining which CAs are added to or removed from the trusted store is perceived as opaque and not terribly uniform across the industry. As mentioned earlier, the CAB Forum has developed baseline requirements for the management and issuance of certificates [CAB2014] for individual CAs. However, the process or set of requirement by which an individual CA gets added to the trust store for each of the major browsers is not clear or transparent. The browser vendors and the CAs determine what should and should not be trusted by default. There are 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 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.

4.2. Dominate Certificate Authorities

The CA industry has evolved to the point where there may be a few very large CAs that are critical to significant portions of Internet infrastructure. The Internet ecosystem must support these CAs; they have become so critical that they cannot be allowed to fail. As a result these CAs are able to dominate the Web PKI. Some mechanism is needed to ensure that these "too big to fail" CAs always act in the best interest of the Internet ecosystem as a whole.

4.3. Cost and Complexity of Deployment

There are many tales about the cost associated with Web PKI deployment and the difficulties that system administrators face to enroll and renew their certificates. The steps required to acquire and install a certificate are both costly and complex. Automation is needed to make these tasks simple and straightforward. In addition, automation should reduce the cost.

5. Emerging Technical Improvements

Many activities are underway to improve the Web PKI. Each of these activities offers some improvement, and taken together, they offer significant improvement.

5.1. Certificate Status Checking

To improve security, certificate status checking needs to be used at all times, either by checking a fresh CRL or by checking an OCSP response. There are ways to make these certificate status checks more efficient.

5.1.1. 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, which can reduce the latency associated with certificate status checking. Sadly, few CAs take advantage of the CRLDP certificate extension.

5.1.2. 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 reduces the number of round trips by the browser to set up the TLS session. 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 simply contact the OCSP responder directly.

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.

5.2. Leveraging the DNS

Two mechanisms leverage the DNS to improve the Web PKI. The first one, DNS-Based Authentication of Named Entities (DANE), provides a verification control that is performed by the browser after the certificate is issued. The second one, Certificate Authority Authorization (CAA), provides an authorization control to be performed by the CA before it issues a certificate.

5.2.1. 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 "mysite.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.

5.2.2. 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.

5.3. Leveraging HTTP

Two mechanisms leverage headers carried in HTTP. The first one, HTTP Strict Transport Security (HSTS), provides notice that all communication with that web server should be TLS-protected. The second one, HTTP Public Key Pinning (HPKP), ensures that the same public keys are used to protect communications with the same server in the future.

5.3.1. 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, which helps protect against eavesdropping and active network attacks. For example, an attacker trying to become a man-in-the-middle has little opportunity to intercept traffic between the browser and the web server.

When a web server includes theStrict-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.

5.3.2. 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 fraudulent certificate issued by a compromised CA.

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.

5.4. Certificate Transparency

Certificate Transparency [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 Certificate Transparency (CT) log.

In the future, a browser may choose to reject certificates without an SCT, and potentially notify the website administrator or CA when they encounter such a certificate. This reporting will help detect mis-issuance of certificates and lead to their revocation.

A 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.

6. Emerging Non-technical Improvements

As with the technical issues around the Web PKI, there are some possible approaches to address the non-technical and process issues.

6.1. Open and Inclusive Web PKI Forum

Is it possible to evolve the CAB Forum or create another forum that would create and manage the Web PKI in an open, inclusive, and transparent manner? In the past, the CAB Forum has decided to remain closed. The CAB Forum Bylaws limit membership to Issuing CAs, Root CAs, and Browser vendors [CAB1.2].

Some organization to manage the Web PKI in an open, inclusive, and transparent manner is needed. This would necessarily includes a process to manage decisions about which CAs are trustworthy. It would be important for this forum to also respond to computer incidents that impact the infrastructure for the Web PKI.

6.2. Cost Effective and Simple Deployment

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 tools for system administrators. The Let's Encrypt project [LE]is already moving in this direction. The expectation is that these protocols and tools will greatly reduce complexity and cost.

7. Security Considerations

TBD.

8. IANA Considerations

None.

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

9. References

9.1. 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, May 2008.

9.2. 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>.

[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", July 2015, <https://letsencrypt.org/>.

[RFC4034] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "Resource Records for the DNS Security Extensions", RFC 4034, March 2005.

[RFC4035] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "Protocol Modifications for the DNS Security Extensions", RFC 4035, March 2005.

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

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

[RFC6698] Hoffman, P. and J. Schlyter, "The DNS-Based Authentication of Named Entities (DANE) Transport Layer Security (TLS) Protocol: TLSA", RFC 6698, August 2012.

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

[RFC6844] Hallam-Baker, P. and R. Stradling, "DNS Certification Authority Authorization (CAA) Resource Record", RFC 6844, January 2013.

[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, June 2013.

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

[RFC6962] Laurie, B., Langley, A., and E. Kasper, "Certificate Transparency", RFC 6962, June 2013.

[RFC7469] Evans, C., Palmer, C., and R. Sleevi, "Public Key Pinning Extension for HTTP", RFC 7469, April 2015.

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 Mary Barnes, Richard Barnes, Marc Blanchet, Alissa Cooper, Nick Doty, Stephen Farrell, Joe Hall, Ted Hardie, Christian Huitema, Eliot Lear, Xing Li, Lucy Lynch, Andrei Robachevsky, Thomas Roessler, Christine Runnegar, 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