Web PKI OPPS
Internet Draft
Intended status: Informational
Expires: December 2014

B. Wilson
Digicert
S. Chokhani
Cygnacom
R. Alden
Comodo

June 9, 2014

Browser processing of server certificates
draft-wilson-wpkops-browser-processing-01.txt

Abstract

This is one of a set of documents to define the operation of the Web PKI. It describes common variations in browser behavior related to processing server certificates.

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 December 9, 2014.

Copyright Notice

Table of Contents

1. Introduction....3
1.1. Definitions....3
1.2. Scope....4
1.3. Document Organization....5
2. Certification Path Development....6
2.1. Basic Requirements....6
2.2. Additional Requirements....7
2.3. Browser/Cryptolibrary Observations....7
2.4. Security Considerations....7
2.5. Areas for Future Work....8
3. Certification Path Validation....8
3.1. Basic Requirements (based on RFC 5280)....8
3.2. Additional Requirements....9
3.3. Browser Observations....9
3.3.1. Path Validation....9
3.3.1.1. Signature Verification....9
3.3.1.2. Name Constraints....9
3.3.2. Current Time within Validity Period....10
3.3.3. Public Key Parameters....11
3.3.3.1. Sizes....11
3.3.3.2. Algorithms and Cipher Suites....11
3.4. Security Considerations....12
3.5. Areas for Future Work....12
4. Server certificate processing....12
4.1. Subject Names....13
4.2. Wildcard character....14
4.3. Key Usage Extension....14
4.4. Security Considerations....15
4.5. Areas for Future Work....15
5. Browser Human Interface (Visual) Indicators....15
5.1. Visual indicators....15
5.2. Positive visual indicators....16
5.3. Negative visual indicators....16

5.4. Message boxes, dialog boxes and error pages.....16
5.5. Certificate viewers.....16
5.6. Certification Path Development and Validation Indication.17
5.7. Configurables.....17

5.8. Security Considerations.....17
5.9. Areas for Future Work.....18
6. IANA Considerations.....18
7. Security Considerations.....18
8. Normative References.....18

1. Introduction

This document reviews the current processing behaviors of cryptolibraries, and the browsers they support, with respect to SSL/TLS session establishment between a server and a browser, including signature verification, certificate parsing, chain processing, and other non-revocation-checking processes described in RFC 5280 and the SSL/TLS protocol.

The information presented in this document is based on user experience and should not be construed as exhaustive. In other words, it is based on observed behavior and is not based on any comprehensive testing. The product vendors and reviewers are encouraged to provide additional information that sheds light on the observations made in this document or to provide additional observations.

This document does not address future changes to the implemented trust model.

1.1. Definitions

PKI terminology is as defined in RFC 5280. Other definitions are defined below for interpretation of this document.

Behavior - The observed action or activity of a browser based on a set of conditions or circumstances.

Blacklist - A group, set, or list of data objects created to explicitly prevent them from being used because they are unsafe or

obsolete.

Block - A behavior in which the browser detects an abnormal condition and halts (or technically cannot complete) session negotiation and drops the connection or otherwise blocks the user from continuing.

Bypassable error - A behavior in which the browser detects an abnormal condition and asks the user whether to proceed with (i.e. click-through to) the SSL/TLS connection.

Fail open - A behavior in which the browser is unable to successfully complete a certificate-checking process, but provides the content.

Internationalized Name - The Punycode-encoded ASCII representation of Unicode characters prefaced with the ASCII Compatible Encoding (ACE) prefix, xn--.

Name mismatch - A condition detected by a browser in which no name in the common name or subject alternative name for the subject in the certificate matches the hostname sought by the client (i.e. the client's reference identity - usually a Fully Qualified Domain Name - is not in the certificate).

Pinned - A condition in which the association between two or more aspects of the entity-public-key relationship (e.g. server name, public key, CA, certificate) are configured and set in the browser before initiation of a TCP connection.

Stapled - A condition in which information related to the server's certificate (e.g. OCSP response) is delivered by the server to the client as part of the SSL/TLS handshake, and not by direct communication with the issuing CA. Not all browsers request stapled responses. Since OCSP stapling is directly related to revocation, discussion of OCSP stapling is outside the scope of this document.

Visual indicator - A behavior in which the browser changes the color(s) and/or intensity of pixels on a screen in the browser chrome to indicate a changed condition. Visual indicators also include error pages, pop-up dialogs, and warning messages.

Wildcard character - An asterisk - * (Unicode 2A).

1.2. Scope

The scope of this document excludes revocation checking. Revocation checking is addressed in another document.

This document currently treats as out-of-scope browser behavior for client authentication TLS. In a future version it might address behavior when a legitimate client certificate is not selected or presented under certain circumstances, such as when the browser and the TLS Server have different trust anchors.

This document also does not address lesser-used X.509 certificate extensions: issuer alternative name; subject directory attribute; policy constraint; inhibit any policy; and subject information access.

Also, because revocation checking is out of scope, the discussion of the following extensions is out of scope: CRL Distribution Point; Freshest CRL; and OCSP field in the Authority Information Access extension. OCSP stapling support is addressed in another document.

This document reviews some of the certificate-processing features of the following cryptolibraries: Network Security Services (NSS), in two code sets, Classic (NSS-Classic) and PKIX (NSS-PKIX); Microsoft Crypto API (MS-CAPI); OpenSSL; and Apple's Cryptographic, Certificate, Key, and Trust Services (A-CKTS). Thus, it examines the behavior of Internet Explorer, on Windows 7/8 relying on MS-CAPI; Mozilla Firefox, relying on NSS; Apple Safari, relying on A-CKTS; and Opera and Google Chrome, relying on MS-CAPI, A-CKTS, or NSS, as the case may be.

Mobile platforms, such as Android, iOS, and Windows Mobile, are also addressed as information becomes available.

1.3. Document Organization

This Section 1 provides a brief introduction to the non-revocation-checking processes implemented in cryptolibraries and by browsers.

Section 2 describes the requirements for certification path development in order to establish trust in the server public key. Section 2 also contains the nuances of cryptolibraries and popular

browsers, in terms of their ability to meet these requirements and security implications of these nuances.

Section 3 describes requirements for certification path validation in order to establish trust in the server public key. Section 3 also contains the nuances of cryptolibraries and popular browsers in terms of their ability to meet these requirements and security implications of these nuances.

Section 4 describes the requirements for processing the Server certificate. Section 4 also contains the nuances of cryptolibraries and popular browsers in terms of their ability to meet these requirements and security implications of these nuances.

Section 5 describes the browser user interface indicators.

Section 6 lists IANA Considerations.

Section 7 summarizes Security Considerations discussed throughout this document.

Section 8 contains references.

2. Certification Path Development

2.1. Basic Requirements

This section discusses sources of certificate information that are available to browsers to assist in certification path development, as described in RFC 5280. The purpose of X.509 path validation is to ensure that the subject distinguished name or subject alternative name in a certificate and the named subject's public key are appropriately bound by a CA that chains up to the public key of a trust anchor used by the browser. The path validation algorithm does this by processing a sequence of certificates that support that binding. While RFC 5280 requires that browsers process certification chains in accordance with the path validation algorithm, it does not specify a procedure by which a browser should construct that certification path. (RFC 4158 provides additional guidance on factors to consider when building a certification path.)

Browsers have or obtain root certificates used to identify the trust

anchors for a server's certification path. Candidate trust anchors are available locally in root stores (maintained by the browser, cryptolibrary, or operating system) or via automatic download from a remote system.

Browsers also use their local caches of certificates for certification path development.

Browser use the certificates sent by the TLS Server in the TLS handshake for certification path development.

Some browsers use the caIssuers field in the Authority Information Access extension of a certificate to obtain, over unsecure HTTP or LDAP, the intermediary CA's certificate in order to build the certification path. Some browsers are able to process LDAP pointers to caCertificate or crossCertificatePair attributes and also handle HTTP single certificate payloads and multiple certificate payloads, as described in RFC 5280.

2.2. Additional Requirements

None

2.3. Browser/Cryptolibrary Observations

All browsers and cryptolibraries examined were able to perform certificate path validation when the server presented the browser with a properly ordered certificate chain, where the first certificate was the end entity's "leaf" certificate, followed by the issuer CA's certificate. However, because a misconfigured server might present a root certificate in the middle of a chain, some cryptolibraries are able to construct certificate paths by re-ordering certificates presented by a server. (There are free tools available to test whether a server is presenting a complete and well-ordered chain.)

Also, however, some servers are misconfigured and only provide the leaf certificate and not a necessary intermediate CA certificate. In these cases, a browser is unable to determine whether the certificate chains to a trusted root. In these cases, the browsers indicate that the site, the connection, or the server is untrusted. Some warning messages indicate that the certificate was not issued by a trusted CA.

Some browsers are able to store intermediate CA certificates permanently or in cache, so they do not need to obtain the intermediate CAs every time.

Some leaf certificates have a caIssuers field in the Authority Information Access extension. The purpose of the caIssuers field is to provide a URI pointer to the Intermediate CA's certificate. All browsers except for Firefox are able to use the caIssuers AIA to obtain the intermediate certificate and construct a chain. Because NSS does not process the caIssuers AIA, Mozilla Firefox is unable to construct a chain. When a path cannot be built, Firefox presents a negative visual indication as a bypassable error as described in Section 5.6. However, Chrome, which uses NSS, has an http-fetching ability it uses to obtain missing intermediate CA certificates.

2.4. Security Considerations

A browser's inability to create a path to a trust anchor leads to uncertainty on the part of end users and may leave them more vulnerable to man-in-the-middle attacks because they are unable to determine whether the public key presented corresponds to the key pair of the server that they intend to reach.

2.5. Areas for Future Work

Inputs are sought from the working group participants and vendors to identify additional methods used and to gain understanding on browser handling of misconfigured server-provided chains when an intermediate CA certificate is available locally to the client.

3. Certification Path Validation

3.1. Basic Requirements (based on RFC 5280)

A browser should only use one or more trust anchors from its root store for certification path validation.

A browser should perform certification path validation in accordance with Section 6 of RFC 5280, including verifying that the signature on the certificate, issuer name, policies, and extensions match given parameters.

Much has been written on the process of signature verification, which requires processing the tbsCertificate and signatureValue fields using the signature algorithm and issuer public key to verify that the certificate was properly signed. A browser should be able to perform the signature verification process using a variety of signature algorithms.

A browser should correctly process the following extensions:

- basicConstraints extension, including the enforcement of path length constraint based on pathLength field in the basicConstraints extension.

- Key usage extension to ensure that the intermediate CA certificates have the certificate signing bit set.

The name constraints extension in a CA certificate should be honored by processing the subject DN and subject alternative names in certificates issued by the CA. (Section 4.2.1.10 and Section 6 of RFC 5280.)

3.2. Additional Requirements

3.3. None. Browser Observations

3.3.1. Path Validation

As mentioned in Section 2.3, online tools can be used to ensure that servers are presenting complete, well-ordered chains, and this should be done to ensure the most efficient certificate path processing. However, when a misconfigured server delivers a shuffled group of certificates, some platforms are unable to perform certification path validation, while other platforms are able to perform validation because they implement a more robust path-building process.

3.3.1.1. Signature Verification

A number of anomalous conditions arise with signature verification

processing: the signature might be plainly erroneous, the signature algorithm might be incorrect, or the browser might not be able to process the algorithm.

When a browser encounters a signature error, it presents an error message such as "invalid signature," "invalid certificate", or "problem with certificate." Browsers exhibit a variety of differing behaviors. For example, Firefox, Chrome, and Opera exhibit a blocking behavior that prevents the user from proceeding to the site. Opera offers a choice between "Back to safety" or "Help me understand". Firefox and Chrome offer "try again" and "reload," as respective options. However, Safari and Chrome on OS X and Internet Explorer on Windows all allow the user to click through this signature validation problem as a Bypassable Error.

3.3.1.2. Name Constraints

We have observed that Chrome and Safari on Mac OSX do not process name constraints. When name constraints are present and marked critical, Chrome presents a message stating, "Something is interfering with your secure connection." However, if the name constraints extension is not marked "critical"," then both Chrome and Safari allow the connection to proceed with no visual indicator of any anomaly. If the name constraint is critical, Safari will reject the certification path due to an unrecognized critical extension (even if the subject DN or the subject alternative name is allowed by the name constraint rule), but it gives the user a choice to proceed with the connection. Chrome on Mac OS X blocks the user with a "reload" button for all name constraints marked critical.

The following additional observations are made with respect to name constraints:

- Microsoft IE on Windows platforms enforces name constraints (in both the CN and in the Subject Alternative Name), but gives the user a choice to proceed with the connection.

- Firefox on all platforms enforces name constraints (in both the CN and in the Subject Alternative Name) and does not permit the user to proceed.

- Chrome on the Windows platform enforces name constraints (in both

the CN and in the Subject Alternative Name), and does not permit the user to proceed.

- Chrome on Linux enforces name constraints in the Subject Alternative Name and does not enforce the name constraint on the CN. Furthermore, in the case of name constraint failure on Linux, Chrome gives the user a choice to proceed with the connection.

3.3.2. Current Time within Validity Period

Most, if not all, browsers properly evaluate whether an end entity certificate is within its stated validity period. The browser may display a warning indicating that a certificate in the certification path is outside of its validity interval or expired. The user may be given the choice to proceed to the content. The trust indicator may be suppressed. In some cases, there may be no warning, but the trust indicator is simply suppressed.

When the browser detects that the current system time is beyond the validity period of a certificate in the certification path, a warning is displayed. Some browsers indicate that a certificate has expired and present a bypassable error asking whether or not to proceed or allowing the user to view the certificate with a certificate viewer. Some browsers also alert the user to the possibility that the error is not caused by an expired certificate, but by incorrect system time, and display the system time. For example, "Your computer's clock currently indicates it is Monday, October 14, 2013, 4:00 AM. Does this look right? If not, you should correct the error and refresh this page."

We plan to enhance this section with additional and more complete information in terms of validity period for certificates in the certification path (i.e., handling expired CA certificates).

3.3.3. Public Key Parameters

3.3.3.1. Sizes

The public exponent for all RSA keys in SSL/TLS certificates must be an odd number and cannot be "1" (because with RSA, when e=1, the cipher text is equal to the plaintext).

Since December 31, 2013, has passed, all public RSA keys should be at least 2048 bits.

Currently, if an RSA key size is less than 900 bits, Opera presents the user with a negative visual indicator and a bypassable dialog. If the RSA key size is greater 900 bits but less than 1,000 bits it removes the padlock indicator.

Microsoft allows manual configuration of minimum key lengths by editing the registry, using a certificate utility or other mechanisms. See http://support.microsoft.com/kb/2661254. Thus, IE and Chrome on Windows platform can enforce the 2,048 bit requirement. It is not known if Firefox on Windows or other browsers listed in the scope section address the 2,048 bit key size requirement.

3.3.3.2. Algorithms and Cipher Suites

Concerns about algorithms and cipher suites have increased over the last several years as new vulnerabilities have been reported in the media. [Examples to be given by cross-reference] This added attention has increased focus on the abilities of browsers to block older, insecure cryptographic methods while at the same time being able to handle and process newer, more secure ones. For instance, Microsoft has announced the deprecation of SHA1 and a 1-January-2017 sunset date of its use in Windows. While other algorithms, like MD4 and MD5, have already been sunsetted, there is the potential that certificates could still be signed using those algorithms. At the same time, use of Diffie-Hellman Ephemeral keys has been suggested as one way to provide forward secrecy, and elliptic curve cryptography is a recommended way to shorten bit lengths of keys. With the various permutations and combinations of parameters for these

algorithms, browser capabilities for the more common ones should be examined.

3.4. Security Considerations

Allowing weak cryptography increases the risk that intercepted communications will be decrypted.

An inability to handle unexpected certificate data may cause a browser to fail in an insecure way. For example, software failure might allow a connection to proceed without encryption, or worse, enable system misuse by malware that exploits the vulnerability.

3.5. Areas for Future Work

Future work will include additional review of algorithm support.

We also plan to discuss how pinning interplays with certification path development and validation. Some browsers support the pinning of public keys.

Most browsers perform certificate policy extension processing appropriately. We have not examined if the policy mapping, inhibit any policy, and policy constraints extensions are processed correctly. Most browsers, as a result of certificate policies extension processing, provide a visual indication when they detect that all the certificates in the certification path contain the correct policy OID for Extended Validation. We plan to quantify further the characteristics for the browsers listed in the scope section.

Inputs are sought from the working group participants and vendors to identify additional path validation rules and additional exceptions.

4. Server certificate processing

This section focuses on how the browsers use names and key usages in Server certificates. While many of these checks are part of certification path validation, these checks are discussed here as part of TLS Server certificate processing in order to emphasize how the browsers use the information in TLS Server certificates.

4.1. Subject Names

SSL/TLS certificates contain at least one subject name to bind the

public key in the certificate with the server that possesses corresponding private key. The subject name appears in the subject alternative name extension as dNSName name type and often in the common name field. The latter practice of using the common name was deprecated by RFC 2818. A browser processes the subject name in the certificate to determine whether it matches the expected server name. (As discussed above, the enforcement of name constraints on the DNS name appearing in certificates varies among browsers.) Browsers are known to successfully connect with servers whose DNS name appears in the Subject CN only and when subject alternative name extension is absent. Current browsers also work if the CN field is blank and the certificate only has the name in the subject alternative name extension. However, of some interest, is confusion over the processing semantics with regard to the "O" and "OU" fields in the subject name. How should subject name be populated when the subscriber is an individual and not an organization? How is the "OU" field interpreted? For EV certificates, what if the CA fails to populate the "O" field?

Browser processing of internationalized names in subject names of certificates allow browsers to either process the Internationalized Name back into Unicode or display the Internationalized Name in ASCII as xn--. See Section 7 of RFC 5280 for more detailed explanation.

In addition to the use of names for SSL/TLS processing, certificate distinguished name fields may provide further identification of the subject through domain-component naming and X.500 naming (e.g. country, organization, etc.). When name constraints are used on the DN, the entire subject distinguished name (not just the CN) needs to pass the name constraints.

Section 3.1 of RFC 2818 states that in the case of a certificate name mismatch, a browser "MUST either notify the user (clients MAY give the user the opportunity to continue with the connection in any case) or terminate the connection with a bad certificate error."

Typical browser behavior will provide a message box that reads, "Security Error: Domain Name Mismatch" with treatment as a bypassable error with options such as "View Certificate," "OK" or "Cancel."

Some browsers still prioritize common name processing over subject alternative name processing even though use of the common name has been deprecated. Another scenario is when the common name is not one of the names listed as a subject alternative name. When either of

these occur, a browser might throw a domain name mismatch even though the name to be used for the SSL/TLS session is in either the common name field or the subject alternative name of the certificate but not in both.

Most browsers display a warning, but allow the user to proceed to viewing the contents of the web site.

Some systems, such as Keychain Access in Apple OS X, allow the user to override certificate name mismatches by explicitly trusting a certificate for a particular domain name that is not contained in the certificate.

4.2. Wildcard character

Most browsers support a wildcard character in the leftmost position. For the browsers listed above in the scope section, we plan to examine wildcard behaviors when the wildcard character is placed in positions other than the leftmost position, or when it alone is located to the immediate left of a top level domain.

4.3. Key Usage Extension

The browsers should use the server public key for key encryption, key agreement or digital signature verification depending on the TLS cipher suite selected. Below are a few examples:

- TLS_RSA_WITH_AES_128_CBC_SHA: The Server key is used for encrypting the master secret and thus, the Server certificate should have the key encipherment bit set if the Server certificate contains the key usage extension.

- TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA: The Server key is used for authentication of ephemeral DH key thus, the Server certificate should have the digital signature bit set if the Server certificate contains the key usage extension.

- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA: The Server key is used for authentication of ephemeral DH key thus, the Server certificate should have the digital signature bit set if the Server certificate contains the key usage extension.

- TLS_ECDH_ECDSA_WITH_3DES_EDE_CBC_SHA: The Server key is used for ECDH key agreement/exchange. Thus, the Server certificate should have the key agreement bit set if the Server certificate contains the key usage extension.

4.4. Security Considerations

An inability to detect and report an improper match between the client's reference identifier (server name) and the subject name in the certificate (a presented identifier) could enable the misuse of a certificate for man-in-the-middle server authentication. For example, an attacker could use a certificate surreptitiously with a server name to bypass a general requirement that the name in the certificate exactly match the fully qualified domain name of the server.

4.5. Areas for Future Work

We plan to add information how the browsers adhere to these requirements. For instance, what is the browser behavior where an elliptic curve certificate asserts the key encipherment bit instead of the key agreement bit?

Also, if the extended key usage extension is present in the Server certificate, it should have one of the following OIDs: Server Authentication or anyExtendedKeyUsage. There is concern about non-SSL/TLS certificates (certificates issued without the intention that they be used for SSL/TLS) with the anyExtendedKeyUsage EKU. We intend to look at the degree of risk this presents.

We plan to add information how browsers vary in their processing of EKUs in end entity certificates.

5. Browser Human Interface (Visual) Indicators

This section describes the typical kinds of browser/OS behaviors when processing SSL/TLS certificates.

5.1. Visual indicators

The most commonly used visual indicator of SSL/TLS security is the padlock icon. Variations of the icon include the closed padlock, the open padlock, and the padlock superimposed with a red slash or X.

5.2. Positive visual indicators

Commonly used visual indicators that are considered positive indications of web site authentication or security are a closed padlock icon, use of the color green, and the display of additional information about issuer or subject of the certificate.

Some of these indicators are called EV indicators because of their use when displaying a website that presents an Extended Validation certificate to the browser.

5.3. Negative visual indicators

Visual indicators used by browsers to convey warnings include use of the color red, a slash (/) or X across a positive indicator (a red slash or X across the padlock icon and/or the "https"), a message box, or the removal of a positive indicator (e.g. removal of the padlock).

5.4. Message boxes, dialog boxes and error pages

A message box is generally used not just as negative indicator, but also to convey more context-specific guidance to the end user. They can provide warnings or explain why an SSL/TLS connection cannot be completed. Dialog boxes are used when the browser encounters an uncertain environmental condition (for gray areas where the security threat is not black or white). Some dialog boxes provide a simple binary choice (a) proceed or (b) "get me out of here." This type of browser behavior can be referred to as a "single bypassable error." Other dialog boxes can exhibit more complex behavior, such as multiple branches, additional nested bypassable errors, helpful information, and decisions to be made by the user.

An error page is another mechanism used by browsers to provide certificate-related information to users.

Some error messages provide an option to view the certificate. Clicking on the offer launches the browser's certificate viewer.

5.5. Certificate viewers

Most browsers provide a means to examine the SSL/TLS certificate of the web site and the chain of certificates leading up to the root certificate. Some browsers block viewing the certificate in circumstances determined by the browser to be insecure.

5.6. Certification Path Development and Validation Indication

If the certification path cannot be validated, some browsers will alert the user about the inability to complete the server's certificate chain, however the clarity of the explanation varies among browsers. For instance, Firefox indicates "connection is unsecure" and that the browser was unable to determine either security status or the identity of the site. It provides the option of "get me out of here" or "I understand the risks".

Most browsers will provide a warning when a certificate is signed by an unknown CA. The warning usually states that an unknown authority issued the certificate. Additional warnings include that if the user has connected to the site previously without errors, it may mean an attacker is trying to impersonate the site and intercept confidential communications. Users are advised not to continue unless they are sure.

With some browsers, this error can be bypassed for the session or the user can explicitly trust the certificate permanently. When a certification path fails because the issuer is not in the root store, most browsers will still allow the user to explicitly trust the certificate or the issuing CA. The number of steps required to explicitly trust an untrusted certificate vary from browser to browser.

5.7. Configurables

Most browsers provide the ability to configure certain certificate-related behaviors. In Mozilla Firefox a user can change some options using Tools -> Options -> Advanced -> Certificates or by typing "about:config" in the address window and editing security preferences. Changes in Microsoft's Internet Explorer settings can be made under Tools -> Internet options -> Advanced -> Security or by editing the registry. In Apple OS X, configuration changes are performed by accessing preferences for certificates in the Keychain, but since the only configurations available are related to revocation checking (CRLs and OCSP), they are outside the scope of this draft.

5.8. Security Considerations

Better (i.e., more accurate, less ambiguous, and more complete) security warnings to end users will lead to better decisions about system security. While there is much to improve with browser error

messages, most operating systems also do not provide information that browsers might use to provide better system diagnoses and messaging.

Better end user messages based on more accurate communication of information can help address concerns about "warning fatigue" and other problems of message ineffectiveness and end user confusion.

5.9. Areas for Future Work

We plan to offer "model" error messages to help guide browsers in communicating with users more clearly.

Inputs are sought from the working group participants and vendors to identify additional problems and solutions.

6. IANA Considerations

This memo includes no request to IANA.

7. Security Considerations

In addition to the security considerations discussed above, the operations described above exhibit several vulnerabilities that could adversely affect the reliability of the authentication and security provided by SSL/TLS certificates. These vulnerabilities have been discussed throughout this RFC and are summarized below:

Additional items will be provided as the draft becomes more stable.

8. References

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

[RFC3647] Chokhani, S., Ford, W., Sabett, R., Merrill, C., and S. Wu, "Internet X.509 Public Key Infrastructure Certificate Policy and Certification Practices Framework", RFC 3647, Nov 2003.

[RFC 4158] Cooper, M., Dzambasow, Y., Hesse, P., Joseph, S.,

Nicholas, R., "Internet X.509 Public Key Infrastructure: Certification Path Building", RFC 4158, September 2005.

[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.

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

[W3C-WSC] Web Security Context: User Interface Guidelines, W3C Recommendation 12 August 2010.

Authors' Addresses

Ben Wilson
Email: ben@digicert.com

Santosh Chokhani
Email: schokhani@cygnacom.com

Robin Alden
Email: robin@comodo.com