Which Web Browsers Process SSL Certificates in a Standardized Way?

Ahmad Samer Wazan[1], Romain Laborde[1], David W. Chadwick[2], François Barrere[1], and AbdelMalek Benzekri[1]

[1] IRIT Laboratory
[2] University of Kent
{wazan,laborde,barrere,benzekri}@irit.fr
d.w.chadwick@kent.ac.uk

Abstract. SSL is the primary technology used to secure web communications. Before setting up an SSL connection, web browsers have to validate the SSL certificate of the web server in order to ensure that users access the expected web site. We have tested the handling of the main fields in SSL certificates and found that web browsers do not process them in a homogenous way. An SSL certificate can be accepted by some web browsers whereas a message reporting an error can be delivered to users by other web browsers for the same certificate. This diversity of behavior might cause users to believe that SSL certificates are unreliable or error prone, which might lead them to consider that SSL certificates are useless. In this paper, we highlight these different behaviors and we explain the reasons for them which can be either a violation of the standards or ambiguity in the standards themselves. We give our opinion of which it is in our analysis.

1 Introduction

The technology used for securing web-based applications is mainly SSL (Secure Socket Layer) [8] or its equivalent standard TLS (Transport Layer security) [9]. SSL relies on X.509 certificates, called here SSL certificates, to provide the confidentiality, authentication, and integrity services for web-based applications.

X.509 certificates are digital identity cards that bind a public key to an entity's name. The entity can be a person, mobile phone, server or any other type of machine. Certificates are issued by Certification Authorities and the X.509v3 standard [6] defines the syntax of these certificates and the semantics of their various fields. Some of the fields are mandatory and some are optional extensions. RFC 5280 [2] refines X.509 for use on the Internet. The Certification Authority (CA) represents the heart of a public key infrastructure (PKI).

Before using a certificate, the relying party[1] (RP) must check whether the certificate is valid or not. The validation process is a complicated task and a multi-risk operation [4]. In the web, executing the validation process by the relying party for each

[1] The entity that relies on the data in the certificate before making its decisions.

D. Gritzalis and J. Lopez (Eds.): SEC 2009, IFIP AICT 297, pp. 432–442, 2009.

SSL connection is impractical for human users. So web browsers execute this process automatically on behalf of them. This implies that web browsers have to be trusted by the users, and consequently, that they should all behave in the same way when processing the same SSL certificate. Web browsers are supposed to conform to the public key standards in order to handle certificates in a uniform way and they should be as transparent as possible from the user's point of view. The experiments presented in this article show that this is not true. Two web browsers might give two different responses for the same certificate. And they often ask complex questions of the user (e.g., "The servers' certificate chain is incomplete, and the signer(s) are not registered. Accept?"). The origin of the differences of behavior is due to either violations of the standards by the browser manufacturers or ambiguity of the standards themselves which leads to multiple interpretations. We give our opinion of which it is in our analysis.

We have tested the latest versions of three popular web browsers (Internet Explorer 7, FireFox 3 and Opera 9.5). The results we obtained have been analyzed to understand the origins of the problems. We have also evaluated the next generation of SSL certificates called Extended Validation certificates (EV certificate) [5] to see if they solve the problems or not. When the cause of inconsistent behavior is the ambiguity of the standards, we propose explicit corrections to the standard to clarify this.

The rest of this paper is structured as follows. Section 2 exposes and analyses the results of tests executed on the three web browsers, and shows why the behaviors of the web browsers are heterogeneous. We also propose remedies to the problems of heterogeneous behavior. In section 3, we discuss the exact role of the relying party. Finally, in section 4 we conclude our study.

2 Analysis of Web Browsers' Behavior

In this section, we provide the results of our tests with Internet Explorer 7 (IE7), Firefox 3 (FF3) and Opera 9.5(OP9) when they validate SSL certificates containing various standard fields. We focus our tests on the fields related to the subject, the key usage and the certificate status. Our approach is to understand the exact meaning applied to these fields by web browsers in web secured communications, by testing their responses when they are confronted with specific test values. The results are analyzed by comparing them to the expected behaviors described in the X.509 standards [6][2]. However, the latter are sometimes ambiguous which may explain the diversity of the browsers' behavior in some cases.

During our experiments, we found three possible responses when web browsers handle SSL certificates, denoted as follows:

- A: accept the certificate without any intervention by the user,
- W: inform the user about the existence of a problem by showing a warning message and asking him/her to take a decision,
- R: refuse the certificate and prohibit the access to the web server without any intervention by the user.

2.1 SSL Certificate Subject

The SSL certificate subject represents the web server. The identity of the server may be either a Fully Qualified Domain Name (FQDN) or an IP address or both. FQDNs and IP addresses are different types of name (called name forms in the standards). A web server could hold many FQDNs that all point to the same IP address, e.g. as in virtual hosting.

2.1.1 What Do the Standards State about the Subject?

The X.509 standard [6] states that the subject field identifies the entity associated with the public-key found in the subject public key field. An entity could have one or more alternative names, of different types (or forms), held in the subjectAltName extension. According to the X.509 standard, an implementation which supports this extension is not required to process all the name types. If the extension is flagged critical, at least one of the name types that is present must be recognized and processed, otherwise the certificate must be considered invalid.

RFC 5280 states that the subject name may be carried in the subject field and/or the subjectAltName extension. If the subject naming information is present only in the subjectAltName extension, then the subject name should be empty and the subjectAltName extension must be critical. According to this statement an SSL certificate can hold multiple names in a combination of the Subject field (CN component) and the Subject Alternative Name (SubjectAltName) extension. These names must all refer to the same entity, although a browser need not recognize all the different name types.

2.1.2 Test and Results

The identity of a server could be represented by a FQDN value or by an IP address or both. We have performed experiments to test certificates holding the two types of name separately as well as both types together.

In the first set of experiments, we tested how the browsers reacted when the certificate contains zero, one or more FQDN names. We configured our web server to respond to requests sent to either www.server1.com or www.server2.com. As the names could be mentioned in either or both of the Subject Name - Common Name (SCN) and SubjectAltName - DNS Name (SAN-DNS) fields, we have tested the following different combinations of names in our web server certificate:

1. SCN=www.server1.com, SAN-DNS=www.server2.com
2. SCN=null, SAN-DNS=www.server2.com
3. SCN=www.server1.com, no SAN-DNS field
4. SCN=null, no SAN-DNS field
5. SCN=null, SAN-DNS = www.server1.com and www.server2.com.

For each combination, we recorded the reaction of each web browser when accessing www.server1.com and www.server2.com (Table 1). We also state whether the certificate is Valid (V) or Invalid (I) according to the X.509 standards. Because we obtained the same results when the SubjectAltName extension was marked critical or not, we haven't indicated this in Table 1.

Table 1. Multiple FQDN Server Identities

	IE7		FF3		OP9		X.509	
By address / Values in fields	S1	S2	S1	S2	S1	S2	S1	S2
i) SCN=S1, SAN-DNS=S2	W	A	W	A	A	A	?	V
ii) SCN=Null, SAN-DNS=S2	W	A	W	A	W	A	I	V
iii) SCN=S1, no SAN-DNS	A	W	A	W	A	W	V	I
iv) SCN=Null, no SAN-DNS	W	W	W	W	W	W	I	I
v) SCN=Null, SAN-DNS=S1,S2	A	A	A	A	A	A	V	V

Where: **S1** = www.server1.com, **S2=** www.server2.com

Table 2. IP Address Server and/or FQDN Identities

IP recognition	IE7		FF3		OP9		X.509	
Accessed by / Values in fields	S1	@IP	S1	@IP	S1	@IP	S1	@IP
i) SAN-IP=192.168.0.6	W	W	W	A	W	W	I	V
ii) SAN-DNS=S1, SAN-IP=192.168.0.6	A	W	A	A	A	W	V	V
iii) SAN-DNS=S1, no SAN-IP	A	W	A	W	A	W	V	I
iv) SAN-DNS=null, SAN-IP=192.168.0.6	W	W	W	A	W	W	I	V

Where: **S1** = www.server1.com, **@IP=**192.168.0.6

In the second set of experiments, we tested how the browsers react when an IP address only, or an IP address and a FQDN, or a FDQN only, are used to identify a web server running at an IP address (with or without the DNS name S1). In all cases the SCN field was null. We obtained the same results when the subjectAltName was marked critical or not, so we have not shown these results in Table 2.

2.1.3 Analysis of the Results

An X.509 certificate binds an identity (the identity of a web server is either a FQDN name or an IP address) to a public key. When the identity of the server is null (Table 1 iv) the browser cannot authenticate the server, so the SSL certificate is invalid. Whether a browser should immediately refuse an invalid certificate (R) or ask the user what to do (W) is partly a usability issue and partly a security issue. But it is not a standard's issue. The standards will only give guidance on whether a certificate is invalid or not, but will not advise a relying party what to do with it. From a security perspective, if the browser cannot authenticate the web server, the certificate should be rejected (R). From a usability perspective the user could be given a choice (W), although in practice most users simply click OK to all the pop up windows so invalid certificates end up being accepted. RFC 5280 mandates that the IP address if present must contain either four (for IPv4) or sixteen (for IPv6) octets, and that the FQDN if

present must not be null. So the Table 1 iv) certificate is clearly invalid. But none of the browsers reject it. Instead they ask the user what to do.

If, the standards are not clear about a certificate's validity, this can lead to web browser implementers holding different interpretations of this. FQDNs should be held in the SAN-DNS extension since this is designed to hold DNS names. However they may also be stored in the common name of the subject distinguished name field (SCN). But what if they are stored in both? [2] states "if the only subject identity included in the certificate is an alternative name form then the subject distinguished name MUST be empty (an empty sequence), and the subjectAltName extension MUST be present." In Table 1, certificate i) appears to violate this rule. But nowhere does the standard explicitly state that such a certificate is invalid; and anyway one can argue that this certificate actually contains two name forms: a subject distinguished name and a SAN DNS name. So this probably explains why IE7 and FF3 treat it as invalid, whilst OP9 treats it as valid. This is why we show a ? in Table 1 i). We have raised the ambiguity of the X.509 standard with ISO/ITU-T and a defect report has been raised and accepted.

[2] says that web browsers must "recognize" the SAN extension, but only that "all parts of the subject alternative name MUST be verified by the CA". This does not place any requirements on the web browser to do likewise. Similarly [6] states "An implementation is not required to be able to process all name forms". So browsers do not have to support SAN-IP, and in fact, IE7 and OP9 do not, so they do not recognise the IP name of the server. FF3 on the other hand does support the IP name form and so does recognise the server's name. This accounts for the different results of Table 2 i), ii) and iv). Whilst all three browsers are still conformant to the standard, they give different results, and a user is not likely to know that this is because the IP name form is not supported by IE7 and OP9.

2.1.4 Do EV Certificates Solve the Problem?

According to the guidelines of the EV certificate, the domain name field can contain one or more host domain name(s) owned or controlled by the subject and be associated with Subject's server. But it doesn't clarify the situation when the identities are held in the CN component and/or in the SAN extension. Also the support of the IP option is not required in this type of certificate. So unfortunately the support for EV certificates will not solve the problems we have identified above.

2.2 Key Usage, Extended Key Usage

Key usage and extended key usage are used to determine the purpose of the public key contained in the certificate. An SSL server certificate could have a key usage extension or not. The standards [2][6] don't constrain the authorities to issue SSL certificates with key usage extensions.

2.2.1 What Do the Standards State about the Key Usage and Extended Key Usage Extensions?

The X.509 standard [6] states that if either the extended key usage or key usage extensions are recognized by the relying party then the certificate must be used just for one of the purposes indicated in the certificate. The key usage and the extended key usage must

be treated separately but they must have consistent values. If there is no purpose consistent with both fields, then the certificate shall not be used for any purpose [6].

RFC 5280 states that the key usage extension, when it appears, should be a critical extension. For an SSL certificate, RFC 5280 recommends that the key usage, when it is defined, should have the value of "digital signature, key encipherment and/or key agreement" and the consistent value of the extended key usage should be "Server Authentication".

The RFC 5280 [2] doesn't restrict any combination of values. The appropriate values for the Key usage extension for particular algorithms are specified in RFC 3279 [7], and other RFCs [2]. For the RSA algorithm, any combination of digitalSignature, nonRepudiation, keyEncipherment and dataEncipherment may be present in the key usage extension [7].

2.2.2 Tests and Results

Technically, the RSA algorithm needs the keyEncipherment value for enciphering the secret keys. Any other value is not needed for the RSA algorithm.

In this experiment, we tested how the web browsers reacted when they validated a certificate which conveyed an RSA public key and had a key usage value different from "keyEncipherment". The same results were obtained when the key usage was

Table 3. Key Usage Test

	IE7	FF3	OP9	X.509
KU=KA and EKU absent	A	W	R	I
KU=DE and EKU absent				I
KU=DE, KA and EKU absent				I
KU=KA and EKU=SA	A	W	R	I
KU=DE and EKU=SA				I
KU=DE, KA and EKU=SA				I
KU=KE and EKU absent	A	A	A	V
KU=KE,DE and EKU absent				V
KU=KE,KA and EKU absent				V
KU=KE,DE,KA and EKU absent				V
KU=KE and EKU=SA	A	A	A	V
KU=KE, DE and EKU=SA				V
KU=KE,KA and EKU=SA				V
KU=KE, DE, KA and EKU=SA				V
KU absent and EKU=CA	R	A	R	I
KU=KE and EKU=CA	R	A	R	I

Where: **KU:** Key Usage extension, **EKU:** Extended Key Usage extension
DE: dataEncipherment, **KE:** keyEncipherment, **KA:** keyAgreement, **CA:** ClientAuth, **SA:** ServerAuth

critical or not, which is correct. We chose wrong values "keyAgreement" and "dataEncipherment" and the correct value "keyEncipherment" as test values for the key usage extension. The final column indicates whether the certificate is valid or invalid according to the standards.

2.2.3 Analysis of Results

Here, the diversity of the web browsers' behaviors is due to violations of the standards when the key usage and/or the extended key usage extension contain wrong values. Certain certificates which should have been treated as invalid were treated as acceptable by IE7 and FF3. OP9 behaved correctly in all the tests and rejected invalid certificates (without asking the user, who is not likely to know anyway). Specifically, IE7 accepted certificates when the key usage had wrong values of data encryption or key agreement instead of key encipherment, and FF3 when the extended key usage had the wrong value of client authentication instead of server authentication. Although not shown in the table, the previous version of Firefox 2 behaves correctly and blocks these accesses. We are not convinced that FF3's behavior in the first six test cases, by asking the user if they wish to use a certificate with an unsuitable key usage value by adding an exception is very helpful, since this will invariably result in an invalid certificate being accepted.

2.2.4 Do EV Certificates Solve the Problem?

The guidelines of the extended validation certificate add new requirements about the presence of the key usage extension for the root certificate and the sub root certificate. For subscriber certificates, EV certificates should follow RFC 5280, so no new requirements are introduced here.

2.3 Revocation

The primary objective of revocation is to remove a non valid certificate from circulation as quickly as possible. This is usually done by asking the relying party to check the certificate's status before accepting it.

Certification authorities can revoke a certificate by either publishing its serial number in a Certificate Revocation List (CRL) that can be downloaded from a repository, or by running a specialized server that can be accessed by the Online Certificate Status Protocol (OCSP) [1]. CrlDistributionPoints *(CDP)* and AuthorityInfoAccess (AIA) extensions are used to hold the CRL and the OCSP indicators respectively in a certificate.

In general, most of the relying parties agreements [e.g. 3] state that relying parties are responsible for taking the risk of using revoked certificates. As a result, relying parties must be aware of the certificate's status before using it in a transaction.

2.3.1 What Do the Standards State about the CRL Distribution Points and Authority Info Access Extensions?

The X.509 standard states that the CDP extension can be, at the option of the certificate issuer, critical or not; but it recommends it to be non-critical for interoperability reasons. When it is a critical extension, the certificate-using systems shall not use the certificate without first retrieving and checking the certificate against the downloaded

CRL [6]. However, when the extension is not critical the certificate-using systems can use the certificate only if the revocation checking is not required by a local policy or it is accomplished by other means [6].

According to RFC 5280, the CDP and AIA extensions should be non-critical extensions, but it recommends supporting these extensions by the authorities and applications [2].

2.3.2 Tests and Results

In the first experiment, we show what are the supported approaches in each web browser and if it is automatically configured or not (Table 4).

Table 4. Supported Approaches

	IE7	FF3	OP9
CRL checking	Automatic	Manual	Automatic
OCSP checking	Automatic	Manual	Automatic

Where: Automatic means that the browser checks the certificate status automatically, and , Manual means that the browser needs to be configured in order to check the certificate status, but once configured checking can be automatic.

In the second experiment (Table 5), we show the reaction of web browsers when the OCSP server is down and checking is automatic.

Table 5. OCSP Server is Down

	IE7	FF2	FF3	OP9
OCSP server is down	A	R	A/R configurable	A

In the third experiment (Table 6), we test the reaction of web browsers when they encounter a certificate signed by an unknown authority.

Table 6. Unknown Authority

Not trusted authority	IE7	FF3	OP9
	W	W	W

Table 7. Certificate is on CRL

	IE7	FF3	OP9
CRL retrieved	R	R	R
CRL not retrieved	A	A	A and degrade

In the fourth experiment (Table 7) we test what happens when we put the certificate serial number on a CRL which is pointed to from a CDP extension, when the CRL can and cannot be retrieved by the browser.

2.3.3 Analysis of Results

The heterogeneity of revocation processes comes from the different implementation efforts by the web browser manufacturers.

Maintaining a revocation service (either CRLs or OCSP) is a requirement for CAs. The standards [2][6] also recommend, but do not mandate, that relying parties ensure that the certificates are not revoked before they rely on them.. However, when the AIA and CDP extensions are present and understood, the relying parties are required to process them. X.509 states about the CDP extension "a certificate-using system shall not use the certificate without first retrieving and checking a CRL from one of the nominated distribution points" Therefore browsers should not ignore these extensions and they should fetch the revocation information and check it before accepting a certificate.

There is some ambiguity over what should happen when a CA says it maintains an OCSP service but does not. RFC 2560 [1] states "the OCSP client suspends acceptance of the certificate in question until the responder provides a response" and "In the event such a connection cannot be obtained, certificate-using systems could implement CRL processing logic as a fall-back position". Thus in the second experiment (Table 5), the responses provided by IE7 and OP9 are not compliant to the standard. Only FF2 and FF3 reject the certificate, although the latest version allows users to configure the browser to accept them. If the browsers cannot fetch the CRL information, then Table 7 shows that none of the browsers are fully conformant as none of them block access, although OP9 removes the padlock icon and asks the user not to send sensitive information.

We conclude that the implementation of the verification mechanisms by the web browsers is weak to say the least, and may allow a relying party to use a revoked certificate without being aware of this:

- Not all web browsers support the automatic verification of certificate status (Table 4)
- When an OCSP server is down the behavior of the browsers is generally not safe (Table 5).
- FF3 updates the CRL list according to the *next update* field of the CRL list. But in reality, nothing prevents a CA from publishing an updated CRL list before the time indicated in this field.
- The relying party may establish a SSL connection with a site without verifying its certificate status and without authenticating the server (Table 6).
- If CRLs are not available the browsers will continue to use the certificate even though they may have been revoked (Table 7).

2.3.4 What Do EV Certificates Say about the Problem?

The guidelines of the EV certificate ask the root authorities to maintain an online 24x7 repository mechanism whereby Internet browsers can automatically check online the current status of all certificates. Conforming CAs must issue a certificate

with either the CDP extension or the AIA extension. However, the guidelines prohibit the CAs from marking these extensions as critical. S

3 Discussion

Sometimes the web browser informs the user of an error in the certificate and asks him/her to take a decision to accept or refuse a connection with the web server, whilst other times the web browser just prohibits the user from accessing the web server or makes the connection immediately even though the certificate is (potentially) invalid. Why are there these conflicting behaviors? Which is the best one? The standards don't answer these questions as they only consider whether a certificate is valid or not. The relying party must make the decision [6] what to do next, but the relying party is sometimes the browser acting on behalf of the user, and sometimes it is the user himself.

Certificate processing should be divided into 2 steps: the validation process (VP) and the decision process (DP). The VP consists of validating the information in a certificate. Most of this processing requires a computer system (e.g. checking a digital signature), some of it requires a human being (e.g. deciding which CA to trust). When the VP process is finished, the DP can choose to accept or not the server's certificate and then make a secure connection, an insecure connection or no connection at all with the server. The latter decision can be based on the certificate's validity and other information (such as failure to get revocation information). However, today, if the browser decides the certificate is valid, it automatically makes the connection without asking the user to decide. If the browser decides the certificate is invalid, then it may decide to send a warning message to the user, and let the user performs the DP, or it may prohibit the user from accessing the web server, and perform the DP itself on behalf of the user. Worse still, occasionally the browser makes the connection automatically using a certificate which it incorrectly decided was valid, without telling the user about this, so that it is opening the user up to potential harm.

If the browser manufacturers had considered the role of the relying party (RP) as two sub-roles, one for the DP and the second for the VP, their behavior could have been more consistent. If a human user performs the DP role and the browser performs the VP role, then the browser cannot either refuse to make a connection or automatically make a connection. The downside of this is that users may get bored with making these decisions and hence always make the connection regardless. If however the browser performs both roles (DP and VP), then the three sets of responses that we see today are possible, and not all browsers will behave in the same way.

4 Conclusions

Which web browser processes SSL certificates in a standardized way? Our experiments have shown that each browser has some non-conformant features. Although the browser implementations were mostly compliant with the standards, occasionally the standards were ambiguous and subject to multiple interpretations which may explain some of the conflicting browser behaviors. Our study was based on an experimental

approach to identifying the non-standard behavior and clarifying the ambiguities in the standards.

The solution to these problems is twofold. Firstly, promoting and clarifying the standards, and secondly, ensuring the web browsers are compliant to these standards. A third approach may be to let the users decide when and if to connect to a web server after the browser informs them of the status of a certificate. Our experiments have led to a defect report on the X.509 standard that has been balloted and accepted. Also our studies show the need for acceptance testing tools to ensure the conformity of web browsers to the standards.

References

1. Myers, M., Ankney, R., Malpani, A., Galperin, S., Adams, C.: X.509 Internet Public Key Infrastructure: Online Certificate Status Protocol – OCSP, RFC 2560 (1999)
2. Cooper, NIST, Santesson, Microsoft, Farrell, Trinity College Dublin, Boeyen, Entrust, Housley, Vigil Security, Polk: Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile, RFC 5280 (May 2008)
3. THAWTE Certification Practice Statement, http://www.thawte.com/en/guides/pdf/Thawte_CPS_2_1.pdf
4. Berbecaru, D., Antonio, L., Marius, M.: On the Complexity of Public-Key Certificate Validation. In: Davida, G.I., Frankel, Y. (eds.) ISC 2001. LNCS, vol. 2200, p. 183. Springer, Heidelberg (2001)
5. CA/Browser forum guidelines for the issuance and management of extended validation certificates, http://www.cabforum.org/EV_Certificate_Guidelines.pdf
6. ITU-T Recommendation X.509 | ISO/IEC 9594-8: Information Technology—Open Systems Interconnection-The Directory: Public-Key and Attribute Certificate Frameworks
7. Polk, W., Housley, R., Bassham, L.: Algorithms and Identifiers for the Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile; RFC 3279 (April 2002)
8. Freier, A., Karlton, P., Kocher, P.: The SSL Protocol Version 3.0, http://wp.netscape.com/eng/ssl3/draft302.txt
9. Dierks, T., Rescorla, E.: The Transport Layer Security (TLS) Protocol Version 1.2. RFC 5246, IETF (August 2008)