Classification of SSL Servers based on their SSL Handshake for Automated Security Assessment

Sirikarn Pukkawanna and Youki Kadobayashi
Nara Institute of Science and Technology
Nara, Japan
Email: {sirikarn-p,youki-k}@is.naist.jp

Gregory Blanc, Joaquin Garcia-Alfaro, and Hervé Debar
Institut Mines-Télécom, Télécom SudParis
Evry, France
Email: {gregory.blanc,joaquin.garcia_alfaro,herve.debar}
@telecom-sudparis.eu

***Abstract*—The Secure Socket Layer (SSL) and Transport Layer Security (TLS) are the most widely deployed security protocols used in systems required to secure information such as online banking. In this paper, we propose three handshake-information-based methods for classifying SSL/TLS servers in terms of security: (1) Distinguished Names-based, (2) protocol version and encryption algorithm-based, and (3) combined vulnerability score-based methods. We also classified real-world SSL/TLS servers, active during July 2010 to May 2011, using the proposed methods. Finally, we propose 45 features, deemed relevant to security assessment, for future SSL/TLS data collection. The classification results showed that servers had bimodal distribution, with mostly good and bad levels of security. The results also showed that the majority of the SSL/TLS servers had seemingly risky certificates, and used both risky protocol versions and encryption algorithms.**

I. Introduction

In today's Internet, the Secure Socket Layer (SSL) and Transport Layer Security (TLS) are the most widely deployed security protocols used when a client and a server desire to securely exchange data over the Internet. SSL/TLS is used in several ways. Online businesses (e.g., online retails) use SSL/TLS to build customers' confidence that their sensitive information will not be compromised during online transactions. Enterprise mail servers utilize SSL/TLS to encrypt messages being transmitted over the Internet or within Intranets.

Unfortunately, as reported by Netcraft in 2012, SSL/TLS can also be used spitefully [28]. Netcraft found a significant number of phishing websites using valid SSL certificates issued by trusted Certificate Authorities (CA), such as Symantec and Comodo. These websites intended to employ HTTPS to convince victims to trust them. Even though they account for a small fraction of phishing attacks, they are eroding trust in SSL/TLS. In the rest of this paper, we will use the term SSL instead of SSL/TLS.

To establish an encrypted communication using SSL, a client and a server perform a handshake. The client requests the SSL certificate from the server. Upon receiving the server's certificate, the client performs certificate verification as follows. It uses the corresponding preloaded CA's public key to verify the authenticity of the digital signature in the server's certificate. It also validates the certificate by checking the certificate's issued and expiry dates. Finally, it generally verifies that the service for which the certificate has been issued matches the service to which it wishes to connect.

The most popular SSL clients and servers are web browsers and servers. Typically, when encountering a server's certificate issued by an untrusted CA, an expired certificate, or a mismatched domain, browsers issue a security warning to users. Browsers also provide more security assistance to their users. For example, by clicking on a padlock in the browser window, users can see information related to the website's certificate, the certificate's issuer, and the period of validity of the certificate. browsers issue a security warning. Furthermore, browsers show a green address bar when a user is connecting to a website using an Extended Validation (EV) certificate. The green bar implies that such a connection is more secure, because CAs use an audited and rigorous entity authentication to issue an EV certificate [37]. However, users must eventually assume the responsibility of trusting an HTTPS website.

The contributions of this paper are fourfold. First, we propose three methods for classifying SSL servers in terms of security: (1) Distinguished Names-based (DN-based), (2) protocol version and encryption algorithm-based, and (3) combined vulnerability score-based methods. Second, we used the proposed methods to classify a large dataset of real-world SSL servers, which were active from July 2010 to May 2011, and present the results. Third, we studied the correlation between the trustworthiness of the certificates and the security of the server-side security parameters. Fourth, we propose 45 features, deemed relevant to security assessment, for future SSL data collection and analysis.

More specifically, the DN-based method checks identity information, called Distinguished Names or DN in the server's certificate, and then uses that information to determine the security level of that server. The protocol version and encryption algorithm-based method directly takes into account known security flaws of the protocol version and the encryption algorithm chosen by the server. The combined vulnerability score-based method examines multiple criteria related to the server-side security parameter settings. It computes security vulnerability scores for each criteria and eventually combines those scores to assess the server. The usefulness of these methods is that they can be implemented as a supplementary client-side security module (e.g., a web browser plug-in) to aid users in assessing the risk of their SSL communications. For example, a web browser integrating our classifier could raise a security alarm when the user's encrypted data may easily be compromised due to a weak cipher, or when the user connects to a malicious server.

The rest of the paper is structured as follows: In Section II,

TABLE I. FEATURES TO DISTINGUISH MALICIOUS CERTIFICATES FROM LEGITIMATE CERTIFICATES [3]

#	Feature name	Type	Notes
1	md5	boolean	The signature algorithm of the certificate is `md5WithRSAEncryption`
2	bogus subject	boolean	The subject section of the certificate is clearly bogus
3	self-signed	boolean	The certificate is self-signed
4	host-common-name-sim	boolean	The subject's common name and domain name share the same domain root
5	issuer common	string	The issuer's common name
6	issuer organization	string	The issuer's organization name
7	issuer country	string	The issuer's country
8	subject country	string	The subject's country
9	validity duration	integer	The validity period (in days)

we describe related work and existing public SSL datasets. In Section III, we describe the SSL handshake along with which information our methods used and also describe the X.509 certificate. In Section IV, we describe the SSL dataset investigated in this paper. We describe the proposed SSL server classification methods in Section V. The classification and study results are shown in Section VI. In Section VII, we discuss the limitations of our methods, emerging SSL handshake-information-based features for security assessment, and present the list of 45 features for future SSL data collection. Finally, we discuss our conclusions in Section VIII.

II. RELATED WORK

A. SSL Server Assessment

There is much research on analyzing and studying trust networks driven by SSL. Coarfa et al. [5] comprehensively studied the performance costs of SSL, such as CPU and cryptographic operational cost. Eckersley and Burns [30], [31] from the Electronic Frontier Foundation (EFF) and iSEC Partners observed several characteristics of Internet X.509 Certificates such as their validity, issuers, and Distinguished Names (DN). Holz et al. [22] studied the X.509 Public Key Infrastructure (PKI), taking particular interest in what cipher and signature algorithms are most widely used, as well as other statistics such as the number of certificates per host and the most prolific certificate issuers between 2009 and 2011. Amann et al. [4] presented a large-scale study of Internet SSL traffic collected passively from five different operational networks. Vratonjic et al. [39] also studied the behavior of SSL certificates, measuring how many SSL servers have domain mismatches. All of them present analysis results that provide a deeper understanding about the current state of the SSL trust network.

In this paper, we try to assess the security level of an SSL server (e.g., HTTPS website) based on the information embedded in the server's handshake responses. This section describes emerging methods that are similar to our work.

As shown in Table I, Almishari et al. [3] proposed a method for detecting web-fraud domains based on nine certificate features. Firstly, they converted certificates to 29-feature vectors which breakdown as follows: features 5 to 8 account for 6 sub-features each, and the remaining features (features 1 to 4, and 9) account for one sub-feature. Secondly, they separately fed the 29-feature vector set to machine learning-based classifiers to train them. Finally, they selected the domains that were labeled as malicious by the best classifier. The best classifier was judged by precision-recall performance metrics. Pan and Ding [33] proposed an anomaly-based method for detecting phishing pages. They considered DN attributes in web certificates as one of several metrics to discriminate phishing pages from legitimate pages. Their assumption is that a phishing page has DN attribute values that are inconsistent with the claimed identity. Ivan Ristic et al. [36] introduced an SSL server rating guideline on behalf of SSL Labs. They proposed assigning a trustworthiness score to SSL servers based on four criteria which are: (1) the SSL protocol version, (2) the key exchange algorithm and the key size, (3) the encryption key size, and (4) the server's SSL certificate. A high score represents high trustworthiness and vice versa. In the guideline, a server will immediately get a score of zero when it has a self-signed, invalid, expired, revoked, or untrusted certificate. If a domain mismatch is found, the server gets a zero score as well. Finally, the scores from all criteria are combined for the final assessment. The OWASP Foundation also provides an SSL server security testing guideline as a part of the OWASP Testing Guide v3 [29]. They state several testing criteria like the SSL Labs criteria, however, they consider two additional features: the data compression and the hashing algorithms. For example, if a server has an X.509 certificate signed using `MD5`, the server is assessed as a vulnerable server due to known collision attacks on this hashing algorithm [40].

B. SSL Data Collection

To the best of our knowledge, the Crossbear project [8] scanned popular websites in the Alexa list [2] from October 2009 to March 2011 from seven locations in the world, namely Germany, China, Russia, Brazil, Australia, Turkey, and the United States. Crossbear's dataset contains information of scanned SSL hosts (such as the host name and the security setting) and their X.509 certificates. The Electronic Frontier Foundation (EFF) Observatory project [17] provides SSL server responses derived by scanning all allocated IPv4 space on the Internet in August and December 2010. The Zmap Team [15] at the University of Michigan [14], [24] and Rapid 7 [23], [35] provide two datasets: SSL certificate and HTTPS Ecosystem. The SSL certificate dataset includes X.509 certificates of servers scanned from October through December 2013. The HTTPS Ecosystem dataset is another comprehensive X.509 dataset collected by performing 110 Internet-wide scans over 14 months between June 2012 and August 2013. Like Crossbear, École Polytechnique Fédérale de Lausanne (EPFL) researchers [39] also provide SSL certificates of the top one million websites ranked by Alexa [2]. During our survey, we noted that most researchers use publicly available datasets but some also create their own, as is the case in [4] where long-term SSL data was collected passively from SSL traffic flowing through operational networks. Active web server scanning is also common as performed by Almishari et al. [3] and for the data used in [25]. SSL datasets from several sources were rarely combined for analysis [21].

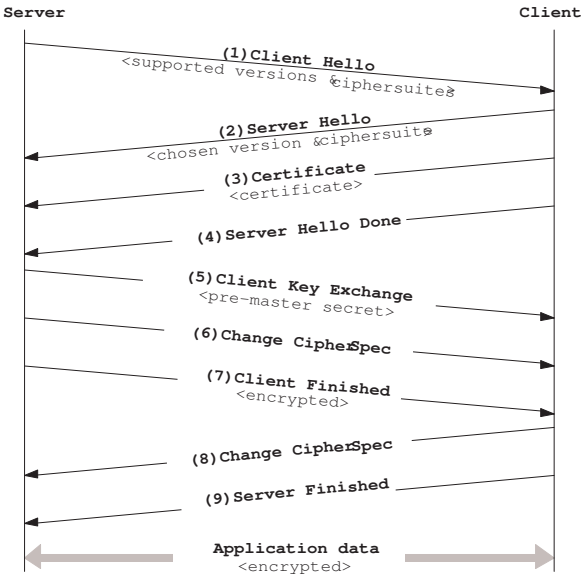

Fig. 1. Steps of the SSL handshake and messages

III. Overview

A. SSL Handshake

SSL helps a client-server application protect the application data during transfer by creating an encrypted channel for private communication over the public Internet. Before an encrypted communication begins, an SSL handshake between the client and server is performed for security parameter negotiation and authentication. This section describes the SSL handshake steps and the information exchanged during the handshake that are used in our work as key information to assess an SSL server. Below we describe the SSL handshake step by step, as shown in Fig. 1.

- The client sends (1) a `Client Hello` message to the server to negotiate security parameters that will be used for the encrypted channel, namely protocol versions, ciphersuites describing key exchange, encryption, hashing algorithms, and compression methods.
- The server chooses an acceptable type for each parameter and replies with (2) a `Server Hello` message.
- The server sends its own SSL certificate, usually an X.509 certificate [9], with (3) a `Certificate` message to the client.
- The server sends (4) a `Server Hello Done` message to notify the client that the server is awaiting a response.
- The client uses the server's certificate to authenticate the server and then sends (5) a `Client Key Exchange` message containing a generated pre-master secret key to the server. Now both the client and the server generate session keys based on the pre-master secret key.
- The client sends (6) `Change Cipher Spec` and (7) `Client Finished` messages to notify the server that the next messages will be encrypted using the session key.
- Finally, the server sends (8) `Change Cipher Spec` and (9) `Server Finished` messages to end the handshake. The client and the server can now exchange application data.

Because our aim was to assess the security level of the SSL server, we examined the information in the server's handshake messages instead of the information in the client's messages. The examined information includes the chosen encryption algorithm in the `Server Hello` message and the certificate in the `Certificate` message because we believe that the encryption algorithm which the server chose is a crucial indicator for assessing security. In addition, we believe that the server's certificate is representative of how secure that server may be.

B. X.509 Certificate

According to the X.509 standard [9], an X.509 certificate contains a public key, a certificate version, a validity period, a serial number, a signature algorithm, and a signature. Aside from those basic fields, the X.509 certificate must contain two Distinguished Names (DN) fields such as the subject DN and the issuer DN. The subject DN is the identity description of the subject who owns the certificate while the issuer DN describes the identity of the Certificate Authority (CA) who issued the certificate. These DNs are used when the client wants to perform name chaining for certification path validation. Technically, the DN is a set of attributes with values separated by comma. For example, a subject DN can be C=US,CN=`www.sample.com` in which the C is the country and the CN the common name which is strictly the domain name. For reasons of compatibility and implementations, as shown in Table II, the certificate must contain six standard DN attributes with qualified values, namely C (Country), O (Organization), OU (Organizational Unit), S (State or province), CN (Common Name), and SerialNumber (Serial number).

IV. Collection of Server Responses

The SSL dataset that was used in this work, consists of five SSL surveys: three private surveys launched in July 2010, April 2011, and May 2011 carried out by Levillain et al. [25]; and two publicly available surveys launched in August and December 2011 by the Electronic Frontier Foundation (EFF) [17]. All surveys consist of response messages derived by probing Internet SSL servers during those periods of time. Table III describes the specification of the `Client Hello` messages that were sent out, the total number of SSL hosts in each survey that answered the `Client Hello` messages, and the total number of SSL hosts categorized by the chosen protocol version. Note that the total number of hosts in the #Total SSL hosts column excludes the number of SSL hosts that replied with `Alert` messages to refuse SSL negotiations. For the Jul-2010, Apr-2011, and May-2011 surveys, to gather the SSL handshake data, Levillain et al. first searched for active Internet IPv4 hosts that were listening on port 443. Second, they sent an SSLv2-compatible `Client Hello` message to each active host and terminated the connection after receiving a `Server Hello Done` message. The `Client Hello`

TABLE II. STANDARD DN ATTRIBUTES AND QUALIFIED VALUES [9]

DN attribute name	Description	Qualified value
C	Country	Any ISO numeric or ISO ALPHA-2 country code that its size does not exceed two or three bytes respectively
O	Organization	Any organization name that its size does not exceed 64 bytes
OU	Organizational Unit	Any organization unit name that its size does not exceed 64 bytes
S	State or province	Any state or province name that its size does not exceed 128 bytes
CN	Common Name	Any absolute domain [32] or Certificate Authority (CA) name that its size does not exceed 64 bytes
SerialNumber	Serial number	Any integer that its size does not exceed 64 bytes

TABLE III. SPECIFICATION OF THE CLIENT HELLO MESSAGES AND THE TOTAL NUMBER OF SSL HOSTS CATEGORIZED BY PROTOCOL VERSION IN EACH SURVEY

Survey name	`Client Hello` message specification		The total number of SSL hosts in survey				
	Offered highest protocol version	Offered ciphersuites	#Total SSL hosts	#SSLv2 hosts	#SSLv3 hosts	#TLSv1.0 hosts	#TLSv1.1 hosts
Jul-2010	TLSv1.0	Standard Firefox suite	9,683,188	0 (0%)	430,973 (4%)	9,252,214 (96%)	1 (<.0%)
Aug-2010	TLSv1.0	SSLv2 and TLSv1.0 suites	11,045,233	11,226 (0.1%)	504,400 (5%)	10,529,606 (95%)	1 (<.0%)
Dec-2010	TLSv1.0	SSLv2 and TLSv1.0 suites	7,705,536	51 (<.0%)	316,933 (4%)	7,388,551 (96%)	1 (<.0%)
Apr-2011	TLSv1.0	Standard Firefox suite	7,134,873	0 (0%)	156,890 (2%)	6,977,983 (98%)	0 (0%)
May-2011	TLSv1.0	Standard Firefox suite	3,796,437	0 (0%)	52,404 (1%)	3,744,133 (99%)	0 (0%)

message for these three surveys offered TLSv1.0 as the highest supported protocol version and offered standard Firefox ciphersuites. For the Aug-2010 and Dec-2010 surveys, EFF sent `Client Hello` messages proposing SSLv2 and TLSv1.0 ciphersuites, and specified TLSv1.0 as the highest supported protocol version. In summary, there were about 9.6 million active SSL servers in the Jul-2010 survey and 11 million SSL servers in the Aug-2010 survey. About seven million SSL servers responded to the Dec-2010 and the Apr-2011 surveys. The May-2011 survey contained about 3.7 million SSL servers. More specifically, most probed SSL hosts (more than 94%) in every survey chose TLSv1.0. The next most popular protocol versions which were chosen were SSLv3 and SSLv2.

V. CLASSIFICATION METHOD

In this paper, we focus on the classification of SSL servers in terms of security using the information embedded in the `Server Hello` and `Certificate` messages sent by the servers during the SSL handshake. This section describes the three proposed classification methods based on that information.

A. Certificate Information

In this work, we make the following assumptions. First, a reliable certificate (e.g., a certificate signed by a trusted CA) tends to include all standard DN attributes with qualified values. Conversely, an unreliable certificate tends to have sloppy DNs. Second, a certificate issued by a known compromised CA or a CA offering incredibly-cheap/free certificates is likely to be a risky certificate. Third, a self-signed certificate is not inherently trusted. Finally, we assume that most malicious servers (e.g., a phishing host) tend to hold unreliable or risky certificates due to their negligence. Based on the information in [9] and these assumptions, we propose the following certificate-based indicators can be used to discriminate risky SSL servers from seemingly harmless SSL servers.

- Indicator 1: at least one standard DN attribute is not presented in the certificate.
- Indicator 2: at least one standard DN attribute value is not a qualified value specified according to Table II.
- Indicator 3: the value of the O/OU attribute contains `self-signed`, `127.0.0.1`, any compromised CA name, or any name of CAs/resellers issuing low-priced or free certificates.

To find compromised CAs, we searched for disclosures regarding CAs that have been compromised and alleged risky CAs. On March 15, 2011, the US CA Comodo reported that its registration authorities had been hacked [7]. In the same year, DigiNotar, a big Dutch CA, was hacked, which resulted in the fraudulent issuing of certificates for a number of domains [38]. GoDaddy was allegedly hacked in 2012, and that resulted in several hours of downtime for millions of websites [27]. However, GoDaddy claimed later that it was only an internal network problem. Below we list the string keywords containing names of the compromised CAs and some CAs/resellers offering low-priced/free SSL certificates identified so far.

- Compromised CAs: `Comodo`, `DigiNotar`, and `GoDaddy`.
- CAs/resellers offering certificates with low-priced costs: `Namecheap`, `RapidSSL`, `fxdomain`, `hostingdude`, and `cheap-domainnames`.
- CAs/resellers offering free certificates: `StartSSL`, `StartCom`, and `CAcert`.

For a real implementation, indicator 2 could be replaced with the following two representative indicators in case that the user's application must reduce memory usage and matching time for analysis. With these representative indicators, the application could use a checklist that is smaller than the size of the full whitelist shown in Table II.

- Representative indicator 1: the CN's value is not in domain name format.
- Representative indicator 2: at least one standard DN attribute (not including SerialNumber) contains one of the following values.
  - an empty or an implied empty value (e.g., "", " ", `NONE`, `None`, `none`, `BLANK`, `blank`, X(s), ?(s), or −(s)),
  - a default value (e.g., S="`SomeState`") or a seemingly meaningless value (see Table IV).

TABLE IV. SEEMINGLY MEANINGLESS VALUES OF EACH STANDARD DN ATTRIBUTE

C (Country)	O (Organization)	OU (Organizational Unit)	S (State/province)	CN (Common Name)
XY, NON-STRING-VALUE, single double quotation	SomeState, Someprovince, SomeOrganization, MyCompany	single double quotation, Single dot, SomeState, Someprovince, SomeOrganizationUnit, Division, section	SomeState, Someprovince, Some_State, Select one, Default, default	localhost.localdomain, 127.0.0.1

TABLE V. PROTOCOL VERSION AND ENCRYPTION ALGORITHM PAIRS FOR SSL SERVER ASSESSMENT

Protocol version	Secure algorithm	Risky algorithm	Insecure algorithm
SSLv3	None	3DES_CBC, IDEA_CBC	DES_CBC, RC2_CBC, RC4
TLSv1.0	None	3DES_CBC, AES_CBC, IDEA_CBC	DES_CBC, RC2_CBC, RC4
TLSv1.1	3DES_CBC, AES_CBC, IDEA_CBC	None	DES_CBC, RC2_CBC, RC4
TLSv1.2	3DES_CBC, AES_CBC, AES_CCM, AES_GCM, Camellia_CBC, Camellia_GCM	None	RC4

B. Protocol Version and Encryption Algorithm

A key assumption we make is that the protocol version and the encryption algorithm chosen by an SSL server are suitable parameters to assess the security level of the communication as well as the SSL server. From a security standpoint, application data sent over a weak protocol version (e.g., SSLv2) or encrypted with a weak encryption algorithm (e.g., RC4 [19]) is in danger due to their security flaws. On the other hand, if the data was sent using a known-good protocol version or a strong encryption algorithm that does not suffer from any known security vulnerabilities, the client can assume that the communication is safer. To assess an SSL server, we analyzed the encryption algorithms supported by each SSL protocol version and assessed their efficiency based on publicly discovered flaws. Table V categorizes the implementation of different encryption algorithms by different versions of SSL with respect to their security. A secure server is one that selects a known-strong protocol version and encryption algorithm. The strongest protocol version at the moment is TLSv1.2. Some known-secure encryption algorithms are the Advanced Encryption Standard (AES) standardized by the US National Institute of Standards and Technology (NIST) in 2001 and Camellia developed later by Mitsubishi and Nippon Telegraph and Telephone (NTT). We consider AES and Camellia as secure encryption algorithms because AES has yet to be compromised by any attacker and Camellia has been proven as strong as AES [34]. Furthermore, to break these algorithms, technically an attacker requires an enormous number of years, e.g., 5×10^{21} years for 128-bit AES encryption [20]. On the other hand, a risky server is one that selects a risky encryption algorithm and SSL protocol version that may be vulnerable to attacks. CBC ciphers are considered risky because they can be broken by the BEAST (Browser Exploit Against SSL) attack [13]. However, if the user's application is designed to defeat the BEAST attack, the attack can be mitigated. For example, Firefox and Chrome using Network Security Services (NSS) libraries for BEAST-like attack mitigation are able to reduce the effect of the BEAST attack. Table V also indicates that if a server uses a newer version of SSL, the security of that server increases because the number of implementation flaws has decreased. For example, a server that chooses 3DES_CBC with TLSv1.1 is described as secure because TLSv1.1 has fixed issues regarding the BEAST attack. Finally, an insecure server is one that selects any publicly known weak encryption algorithm. For example, DES has been defeated by brute-force and differential [16] attacks, and RC2 or RC4 have been defeated by related-key attack [18]. Most importantly, a server that selects old-fashioned SSLv2 with any encryption algorithm is immediately considered as insecure because SSLv2 is flawed in a variety of ways [1]. For example, it has a weak Message Authentication Code (MAC) construction that uses MD5 with a secret prefix, making it vulnerable to length extension attacks [12].

C. Security Assessment Score

Instead of using a single criterion to assess the security of an SSL server, we combine two criteria, still based on the server response: (1) the SSL protocol version and the encryption algorithm that the server chooses and (2) the certificate's DN. Table VI shows the assessment criteria with cases and the assigned scores, which are vulnerability scores, of each case. Criteria 1 consists of nine cases ranked from high to low vulnerability based on the known flaws of those protocol versions and encryption algorithms similar to the proposed method in Section V-B. In Table VI, the set of protocol version and encryption algorithm pairs (secure, risky, and insecure) is the same set shown in Table V. For the scores of criteria 1, we assigned high scores for high vulnerability cases and assigns low scores for low vulnerability cases. As a result, we initially assigned the highest score, which is 1.5, to case 1.1 associated with SSLv2. Due to the smaller number of security flaws of SSLv3 compared to SSLv2, cases 1.2 to 1.4 were assigned lower scores which are in a range from 1.25 to 1. The scoring gap among those SSLv3 cases is 0.125. For the same reason, cases 1.5 to 1.7 associated with TLSv1.0 were assigned a range with lower scores that are between 0.625 to 0.5. The scoring gap between the TLSv1.0 cases is smaller than the scoring gap of SSLv3 cases. It is 0.625, which is one half of the scoring gap of SSLv3 cases (0.125/2). For case 1.8 associated with TLSv1.1, we gave a score of 0.0625, which is smaller than case 1.7's score. We assigned the lowest score to case 1.9 associated with TLSv1.2, which is the most secure SSL protocol at the moment. As a result, case 1.9 has a score of 0.03125, which is one half of case 1.8's score. Next, criteria 2 consists of two cases associated with the certificate's DN. Case 2.1 conforms to indicator 1 described in Section V-A. Case 2.2 conforms to indicators 2 and 3. We assigned a score of 1 to case 2.1, which is similar to the score of case 1.4 because we assumed that a server with a certificate not conforming to the X.509 standard has a moderate vulnerability. Finally, case 2.2 was assigned

TABLE VI. CRITERION AND VULNERABILITY SCORES FOR SSL SERVER ASSESSMENT

Criteria 1: vulnerability of protocol version & encryption algorithm	**Vulnerability score**
1.1: SSLv2 with any encryption algorithm	1.5
1.2: SSLv3 with a weak encryption algorithm	1.25
1.3: SSLv3 with a risky encryption algorithm	1.125
1.4: SSLv3 with a secure encryption algorithm	1
1.5: TLSv1.0 with a weak encryption algorithm	0.625
1.6: TLSv1.0 with a risky encryption algorithm	0.5625
1.7: TLSv1.0 with a secure encryption algorithm	0.5
1.8: TLSv1.1 with a weak encryption algorithm	0.0625
1.9: TLSv1.2 with a weak encryption algorithm	0.03125
Criteria 2: trustworthiness of certificate's DN	**Score**
2.1: At least one standard DN attribute does not presented in the certificate	1
2.2: A standard DN attribute with invalid value	0.2/instance

TABLE VII. CLASSIFICATION RESULTS BASED ON CERTIFICATE INFORMATION

Survey name	#Seemingly harmless servers	#Risky servers
Jul-2010	3,291,377 (34%)	6,391,811 (66%)
Aug-2010	3,749,160 (34%)	7,296,073 (66%)
Dec-2010	2,957,136 (38%)	4,748,400 (62%)
Apr-2011	2,193,934 (31%)	4,940,939 (69%)
May-2011	1,149,757 (30%)	2,646,680 (70%)

a score of 0.2 for each found invalid value. For example, if the C (Country) attribute is not a country code, and the OU (Organizational Unit) attribute contains self-signed, the total score of case 2.2 is 0.4. Note that these scores are adjustable as long as those scores are in the same order with the score in Table VI or each score is scaled accordingly. For example, if a score of 10 is assigned to case 1.1, then case 1.2's score is supposed to be less than 10. In addition, case 1.3's score is supposed to be less than case 1.2's score.

To assess the security level of an SSL server, the total vulnerability score ($Tscore$) derived from combining criteria 1's score and criteria 2's score is calculated. Then, the $Tscore$ is used to determine the security level of that server. More specifically, the $Tscore$ is calculated using the following equation.

$$Tscore = \omega_1(criteria1Score) + \omega_2(criteria2Score) \quad (1)$$

where $criteria1Score$ is the score of criteria 1 and $criteria2Score$ is calculated by summing the scores of case 2.1 and case 2.2. The ω_1 and ω_2 are weights for criteria 1 and 2 respectively.

VI. EXPERIMENTAL RESULTS

We performed three experiments to classify real-world SSL servers based on the three proposed methods described in Section V. We used the five SSL surveys from Levillain et al. and the Electronic Frontier Foundation (EFF) described in Section IV. Table III shows their details.. We also studied the behavior of the SSL servers in those surveys, particularly focusing on the relationships between: (1) the certificate quality and the protocol version and (2) the certificate quality, the cipher strength, and the trustworthiness of key exchange algorithm. This section describes the classification results of each method and the study results in detail.

A. Certificate Information

In this experiment, we clustered the SSL servers in the surveys based on the certificate-based classification method that we proposed in Section V-A. We used indicators 1 and 3, and the representative indicators 1 and 2 to separate the SSL servers into two classes: risky and seemingly harmless. We used the same keyword set of compromised CAs and CAs/resellers described in Section V-A for indicator 3. Note that in this experiment, we inspected only the certificate's subject DN because we lacked information of the issuer DN for the Jul-2010, Apr-2011, and May-2011 surveys. If a server's certificate matches at least one indicator, the server is classified as risky immediately. Table VII shows the total numbers of seemingly harmless and risky SSL servers in each survey. The results indicate that more than 61% of the SSL servers in every survey fell into the risky class and the rest of the servers are in the seemingly harmless class. This means that most SSL servers, active during July 2010 to May 2011, had subject DNs that did not conform to the X.509 standard and/or contained seemingly risky values.

TABLE VIII. CLASSIFICATION RESULTS BASED ON PROTOCOL VERSION AND ENCRYPTION ALGORITHM

Survey name	#Secure servers	#Risky servers	#Insecure servers	#Unknown servers
Jul-2010	0 (0%)	5,972,246 (62%)	3,400,572 (35%)	310,370 (3%) (3%)
Aug-2010	1 (<0.5%)	7,068,241 (64%)	3,970,174 (36%)	6,817 (<0.5%)
Dec-2010	1 (<0.5%)	4,938,107 (64%)	2,762,196 (36%)	5,232 (<0.1%)
Apr-2011	0 (0%)	4,444,114 (62%)	2,350,419 (33%)	340,340 (5%)
May-2011	0 (0%)	3,675,969 (97%)	109,409 (3%)	11,059 (<0.5%)

B. Protocol Version and Encryption Algorithm

The purpose of this experiment was to classify the SSL servers in the surveys based on the SSL protocol versions and the encryption algorithms that the servers chose during handshakes. We used Table V to classify the servers into three classes: secure, risky, and insecure. If a server chose an encryption algorithm that is not contain in Table V, we classified it as an unknown server. Table VIII shows the total numbers of SSL servers classified into each class for each survey. The results indicate that more than 61% of the SSL servers in every survey appear to be risky. Interestingly, up to 97% of the servers in the May-2011 survey chose risky protocol version and encryption algorithm pairs. In this work, we did not check to see if those risky servers that we found in the May-2011 survey existed or not in the older surveys (the Jul-2010, Aug-2010, Dec-2010, and Apr-2011 surveys). Thus, we could not confirm whether the servers tried to downgrade the security. The results also indicate that there was only one secure server in the Aug-2010 and Dec-2010 surveys. Except for the May-2011 survey, about 35% of all servers chose insecure protocol version and encryption algorithm pairs. Finally, we encountered a few servers that used unknown encryption algorithms.

C. Security Assessment Score

Next, we measured the security levels of the SSL servers using the score-based method proposed in Section V-C. For case 2.2, we used the representative indicators 1 and 2. In each survey, by using equation 1 and the assigned scores in Table VI, we calculated the $Tscore$ of each server. In this

TABLE IX. VULNERABILITY SCORE RANGES WITH SECURITY LEVELS TO ASSESS AN SSL SERVER

Total vulnerability score ($Tscore$) range	Security level
$Tscore < 0.5$	Best
$0.5 \leq Tscore < 1$	Good
$1 \leq Tscore < 1.5$	Average
$1.5 \leq Tscore < 2$	Bad
$Tscore \geq 2$	Worst

experiment, ω_1 and ω_2 were set to be one, which means that we have weighted the two criteria equally. As a result, the $Tscore$ was in the range from 0 to 3.5 ($1(1.5)+1(1+(5\times0.2))$). Note that the values assigned to ω_1 and ω_2 can be adjusted, depending on how much importance is given to each criteria. For example, if using a weak protocol version/encryption algorithm is considered more critical than having an untrusted DN, then $\omega_1 > \omega_2$. Instead of directly representing the security level of a server by its $Tscore$, we defined five intuitive terms to qualitatively describe the security levels: (1) best, (2) good, (3) average, (4) bad, and (5) worst. Table IX shows the total vulnerability score ($Tscore$) ranges for the five security levels. The classification results based on their total vulnerability scores are shown in Table X. The results reveal that there are no servers that appear to be the best server in terms of security in the surveys. About 43% of the servers are good servers and about 50% of the servers are bad servers, a bimodal distribution. The results also show that less than 7% of the servers have either average or worst security levels.

D. SSL Server Behavior

To gain more knowledge from the SSL dataset, we also studied the SSL server behavior. Our aim was to investigate the relationships between (1) the certificate quality and the protocol version; and (2) the certificate quality, the cipher strength, and the security level of the key exchange algorithm of the SSL servers in the surveys.

1) Certificate quality and protocol version: There are three grades of SSL certificates in general, namely Extended Validation (EV), Organization Validation (OV), and Domain Validation (DV). The EV certificate is widely considered to be the most trusted SSL certificate nowadays because obtaining it requires extensive entity verification of the certificate requester by a Certificate Authority (CA) [37]. Thus, an SSL server holding an EV certificate is fairly reliable. The OV certificate requires less entity verification steps than the EV certificate. The DV certificate provides the lowest level of entity verification because a DV certificate can be issued online and often is offered at a much lower price than the OV and EV certificates. This makes the DV certificate less trustworthy than the OV and EV certificates. Self-signed certificates are another type of SSL certificates which are issued by the server themselves with no entity verification. Thus, a self-signed certificate is unreliable.

In this study, we assessed the quality of the SSL certificates in the surveys based on their certificate type and subject DN. We found that most SSL servers in the surveys held OV certificates and provided imperfect DNs. Furthermore, surprisingly we found that some servers chose TLSv1.1 and had self-signed certificates. This means that an SSL server that chooses a secure protocol version does not necessarily also use a trusted certificate.

TABLE X. CLASSIFICATION RESULTS BASED ON SECURITY ASSESSMENT SCORE

Survey name	#Good servers	#Average servers	#Bad servers	#Worst servers
Jul-2010	4,462,945 (46%)	472,745 (5%)	4,494,127 (46%)	253,371 (3%)
Aug-2010	4,933,374 (42%)	521,230 (5%)	5,307,106 (48%)	283,523 (3%)
Dec-2010	3,720,084 (48%)	308,675 (4%)	3,482,081 (45%)	194,696 (3%)
Apr-2011	2,764,267 (39%)	258,491 (4%)	4,027,915 (56%)	84,200 (1%)
May-2011	1,526,521 (40%)	212,406 (6%)	2,053,710 (54%)	3,800 (<0.5%)

2) Certificate quality, cipher strength and trustworthiness of key exchange algorithm: Another aim was to study the relationship between the certificate quality, the cipher strength, and the trustworthiness of the key exchange algorithm that SSL servers in the surveys chose during a handshake. To study this, the certificate quality of a server was measured based on the certificate type and the subject DN similar to the previous study. For cipher strength, we used the information shown in Table V as well as the key size that the server used to encrypt the data. Fundamentally, a large key size will have more strength than a small key size because an attacker requires more time to compromise such key. For example, using the same encryption algorithm, if data A is encrypted with a 256-bit key and data B is encrypted with a 128-bit key, then data A is safer than data B. The last feature that we studied was the trustworthiness of the key exchange algorithm identified in the chosen ciphersuite in the Server Hello message. This algorithm is used in the authentication process between a client and a server when they do a handshake. To assess the trustworthiness of the key exchange algorithm, we simply assumes that an anonymous key exchange algorithm is likely to be less robust. On the other hand, a well-known key exchange algorithm is likely to be more trustworthy. Our study found that the majority of SSL servers in every survey chose well-known key exchange algorithms, such as RSA, DHE_RSA, DHE_DSS, DH_RSA, and DH_DSS. Furthermore, we found that some SSL servers in the surveys had unreliable SSL certificates although they also chose very strong ciphers and vice versa.

VII. DISCUSSION

By analyzing the collection surveys, we found that most SSL servers chose risky encryption algorithms (e.g., RC4) with well-known key exchange algorithms. This confirms what Holz et al. [22] and Amann et al. [4] stated who discovered that the most frequent cipher used is RC4 with RSA. We also found that most servers in the surveys were holding Organization Validation (OV) certificates, which are in practice more trustworthy than Domain Validation (DV) and self-signed certificates.

Our results were based solely on known flaws which have been disclosed, among the current implementations on the server side. Therefore, for the moment, our methods may provide effective and precise assessments until a new flaw/attack on cryptographic protocol is revealed or its protocol implementation is refined. This is one limitation of our methods. To preserve the efficiency and accuracy of assessment of our methods, the classification models must be updated regularly.

TABLE XI. PROPOSED 45 FEATURES FOR FUTURE SSL SERVER ASSESSMENT

#	Feature name	Type	Feature description
1	Country	string	The county code where the server is located
2	City	string	The city name where the server is located
3	Region	string	The region code where the server is located
4	Area	integer	The area code where the server is located
5	Time zone	string	The time zone of the region where the server is located
6	DMA	integer	The Designated Market Area code where the server is located
7	Metro	integer	The metropolitan area code where the server is located
8	Postal code	integer	The postal code where the server is located
9	Latitude	number	The latitude number where the server is located
10	Longitude	number	The longitude number where the server is located
11	Continent	string	The continent name where the server is located
12	Organization	string	The organization name who owns the server
13	AS	integer	The Autonomous System number of the zone where the server is located
14	ISP	string	The Internet Service Provider of the server
15*	Protocol version	number	The SSL protocol version that the server chooses during SSL handshake
16	Key exchange algorithm	string	The key exchange algorithm that the server chooses during SSL handshake
17	Hashing algorithm	string	The hashing algorithm that the server chooses during SSL handshake
18*	Encryption algorithm	string	The encryption algorithm that the server chooses during SSL handshake
19*	X.509 country	string	The country attribute's value specified in the SSL certificate's Distinguished Names (DN)
20*	X.509 state	string	The state attribute's value specified in the SSL certificate's DN
21	X.509 city	string	The city attribute's value specified in the SSL certificate's the SSL certificate's DN
22*	X.509 organization	string	The organization name attribute's value specified in the SSL certificate's DN
23*	X.509 organizational unit	string	The organizational unit attribute's value specified in the SSL certificate's DN
24*	X.509 common name	string	The common name attribute's value specified in the SSL certificate's DN
25	X.509 domain component	string	The domain component attribute's value specified in the SSL certificate's DN
26	X.509 surname	string	The surname attribute's value specified in the SSL certificate's DN
27	X.509 given name	string	The given name attribute's value specified in the SSL certificate's DN
28	X.509 email address	string	The email address attribute's value specified in the SSL certificate's DN
29	X.509 MAC	string	The Message Authentication Code attribute's value specified in the SSL certificate's DN
30	X.509 serial number	number	The serial number attribute's value specified in the SSL certificate's DN
31	X.509 title	string	The title attribute's value specified in the SSL certificate's DN
32	X.509 description	string	The description attribute's value specified in the SSL certificate's DN
33	X.509 business category	string	The business category attribute's value specified in the SSL certificate's DN
34	X.509 postal address	string	The postal address attribute's value specified in the SSL certificate's DN
35	X.509 postal code	string	The postal code attribute's value specified in the SSL certificate's DN
36	X.509 post office box	string	The poster office box number attribute's value specified in the SSL certificate's DN
37	X.509 street address	string	The street address attribute's value specified in the SSL certificate's DN
38	X.509 telephone number	number	The telephone number attribute's value specified in the SSL certificate's DN
39	X.509 initials	string	The initials attribute's value specified in the SSL certificate's DN
40	X.509 certificate type	string	The type of the server's certificate (e.g., Extended Validation)
41	Security degree of protocol version	number	The vulnerability score of the SSL protocol version that the server chooses during SSL handshake
42	Security degree of X.509 certificate	number	The vulnerability score of the server's SSL certificate
43	Security degree of cipher	number	The vulnerability score of the encryption algorithm and encryption key size that the server chooses
44	Security degree of key exchange mechanism	number	The vulnerability score of the key exchange mechanism that the server chooses
45*	Security degree of SSL server	number	The overall vulnerability score of the server

Aside from the SSL protocol version, our work considered the cryptographic parameters (encryption algorithm and its key size), the key exchange algorithm, and the server's certificate (type and its identity description). However, other entities also can be used as indicators to measure the security level of an SSL communication/server. SSL Labs [36] assumed that an SSL certificate that has been revoked (whatever the reason) should not be trusted, thus a server with a revoked certificate is penalized as an insecure server. Currently, some web browsers (e.g., Internet Explorer from version 7 and Firefox) optionally consider the revocation status of an X.509 certificate to help their users verify the certificate's validity by using the Online Certificate Status Protocol (OCSP). In [36], the authors also argue that a well-deployed SSL server should avoid the use of a wildcard certificate. They claim that although existing wildcard certificates are not any less secure from a strict technical point of view, the way in which these wildcard certificates are typically handled (especially in larger organizations) makes them less secure in practice. Furthermore, using a wildcard is not permitted for an EV certificate. Thus, the existence of a wildcard may be an good indicator for SSL security assessment. The hashing algorithm used to create the message digest can also be used for SSL server security rating [29], [36]. The server's public key can be used as well to determine the likelihood of exploitation of an SSL server using an old version of OpenSSL [11]. The cryptographic library/toolkit that the server uses can be used for assessment. For example, if an SSL server uses OpenSSL version 1.0.1 through 1.0.1f or 1.0.2 beta through 1.0.2-beta1, it is vulnerable to the Heartbleed attack [6]. The data compression algorithm for an SSL channel is also a relevant indicator [29] because if compression is enabled, the communication can be compromised by the CRIME and BREACH attacks [10], [26]. As a result, most web browsers currently either disable or permanently remove the compression feature to mitigate these attacks.

Finally, based on the experiences gained through this study, we propose 45 features shown in Table XI for future SSL data collection. These features consist of the features or indicators used in the research and practice described above as well as the geolocation features of SSL servers deemed relevant to security assessment. Note that the eight features marked with "*" were used in this work and the remaining features are the newly proposed features. More specifically, features 1 to 14 are the geolocation information of the server, such as the country and the city where the SSL server is located, and the Internet Service Provider (ISP) of the server. Feature 15 is

the protocol version chosen by the server. Features 16 to 18 are security algorithms described in the chosen ciphersuite. Features 19 to 40 are the certificate's DN attributes, which include standard and optional DN attributes. Features 41 and 42 are the vulnerability scores of the chosen SSL protocol version and the server's certificate respectively. Feature 43 is the vulnerability score of the cipher (encryption algorithm and encryption key size). Feature 44 is the key exchange mechanism's vulnerability score. Finally, feature 45 is the overall vulnerability score of the server

VIII. Conclusion

In this paper we proposed three methods to classify SSL servers in terms of security: (1) Distinguished Names-based (DN-based), (2) protocol version and encryption algorithm-based, and (3) combined vulnerability score-based methods. Then we classified Internet SSL servers active between July 2010 and May 2011 based on our proposed methods. We found that more than 61% of the SSL servers were classified as risky because they were using SSL certificates containing seemingly meaningless subject DNs and/or choosing risky SSL protocol versions and encryption algorithms for communications. By considering multiple criteria, we found servers had a bimodal distribution, with mostly good and bad levels of security. Furthermore, we studied a correlation between the trustworthiness of the certificates and the security of the security parameters that the servers chose. We did not find a correlation between them: a server with a trusted certificate may provide insecure communication and vice versa. Finally, we also found that the majority of the servers had Organization Validation (OV) certificates.

Acknowledgment

This research has been supported by the Strategic International Collaborative R&D Promotion Project of the Ministry of Internal Affairs and Communication, Japan, and by the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement No. 608533 (NECOMA). The opinions expressed in this paper are those of the authors and do not necessarily reflect the views of the Ministry of Internal Affairs and Communications, Japan, or of the European Commission.

References

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

[2] Alexa. The top 500 sites on the web. Available at: http://www.alexa.com/topsites.

[3] M. Almishari, E. De Cristofaro, K. El Defrawy, and G. Tsudik. Harvesting SSL Certificate Data to Identify Web-Fraud. *I. J. Network Security*, 14(6):324–338, 2012.

[4] B. Amann, M. Vallentin, S. Hall, and R. Sommer. Revisiting SSL: A Large-Scale Study of The Internet's Most Trusted Protocol. Technical report, ICSI, December 2012.

[5] C. Coarfa, P. Druschel, and D. S. Wallach. Performance Analysis of TLS Web Servers. *ACM Trans. Comput. Syst.*, 24(1):39–69, Feb. 2006.

[6] Common Vulnerabilities and Exposures. CVE-2014-0160. Available at: https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2014-0160.

[7] Comodo. Report of incident on 15-mar-2011. Technical report, March 2011. Available at: https://www.comodo.com/Comodo-Fraud-Incident-2011-03-23.html.

[8] Crossbear Project. SSL Landscape - X.509 data sets. Available at: https://pki.net.in.tum.de/node/8.

[9] D. Cooper, S. Santesson, S. Farrell, S. Boeyen, R. Housley, and W. Polk. Internet X.509 Public Key Infrastructure Certificate and CRL Profile. RFC5280.

[10] D. Goodin. Crack in Internet's foundation of trust allows HTTPS session hijacking. *Ars Technica*, September 2012. Available at: http://arstechnica.com/security/2012/09/crime-hijacks-https-sessions.

[11] Debian. DSA-1571-1 openssl – predictable random number generator, 2008. Available at: http://www.debian.org/security/2008/dsa-1571.

[12] T. Duong and J. Rizzo. Flickr's API Signature Forgery Vulnerability. Technical report, September 2009. Available at: http://netifera.com/research/flickr_api_signature_forgery.pdf.

[13] T. Duong and J. Rizzo. Here Come The ⊗ Ninjas. May 2011. Unpublished manuscript.

[14] Z. Durumeric, J. Kasten, M. Bailey, and J. A. Halderman. Analysis of the HTTPS Certificate Ecosystem. In *Proceedings of the 2013 Conference on Internet Measurement Conference*, IMC '13, pages 291–304, New York, NY, USA, 2013. ACM.

[15] Z. Durumeric, E. Wustrow, and J. A. Halderman. ZMap - The Internet Scanner. Available at: https://zmap.io.

[16] E. Biham and A. Shamir. Differential Cryptanalysis of DES-like Cryptosystems. *Cryptology*, 2(3):3–72, July 1990.

[17] Electronic Frontier Foundation. The EFF SSL Observatory. Available at: https://www.eff.org/observatory.

[18] B. Eli. New types of cryptanalytic attacks using related keys. *Cryptology*, pages 229–246, 1994.

[19] S. Fluhrer, I. Mantin, and A. Shamir. Weaknesses in the Key Scheduling Algorithm of RC4. In *in Proceeding of the 4th Annual Workshop on Selected Areas of Cryptography*, SAC'1, pages 1–24, 2001.

[20] H. O. Alanazi, B. B. Zaidan, A. A. Zaidan, H. A. Jalab, M. Shabbir and Y. Al-Nabhani. New Comparative Study Between DES, 3DES and AES within Nine Factors. *Computing*, 2(3):152–157, March 2010.

[21] N. Heninger, Z. Durumeric, E. Wustrow, and J. A. Halderman. Mining Your Ps and Qs: Detection of Widespread Weak Keys in Network Devices. In *Presented as part of the 21st USENIX Security Symposium (USENIX Security 12)*, pages 205–220, Bellevue, WA, 2012. USENIX.

[22] R. Holz, L. Braun, N. Kammenhuber, and G. Carle. The SSL Landscape: A Thorough Analysis of the x.509 PKI Using Active and Passive Measurements. In *Proceedings of the 2011 ACM SIGCOMM Conference on Internet Measurement Conference*, IMC '11, pages 427–444, New York, NY, USA, 2011. ACM.

[23] Internet-Wide Scan Data Repository. Rapid 7 - SSL Certificates. Available at: https://scans.io/study/sonar.ssl.

[24] Internet-Wide Scan Data Repository. University of Michigan - HTTPS Ecosystem Scans. Available at: https://scans.io/study/umich-https.

[25] O. Levillain, A. Ébalard, B. Morin, and H. Debar. One Year of SSL Internet Measurement. In *Proceedings of the 28th Annual Computer Security Applications Conference*, ACSAC '12, pages 11–20, New York, NY, USA, 2012. ACM.

[26] J. Leyden. Step into the BREACH: HTTPS encrypted web cracked in 30 seconds. *The Register*, August 2013. Available at: www.theregister.co.uk/2013/08/02/breach_crypto_attack.

[27] E. Mills. Go Daddy-serviced Web sites go down; hacker takes credit. *CNET*, September 2012. Available at: http://www.cnet.com/news/go-daddy-serviced-web-sites-go-down-hacker-takes-credit.

[28] Netcraft. Phishing Alerts for SSL Certificate Authorities. Available at: http://news.netcraft.com/archives/2012/08/22/phishing-on-sites-using-ssl-certificates.html.

[29] Open Web Application Security Project. Testing for SSL-TLS (OWASP-CM-001). Available at: https://www.owasp.org/index.php/Testing_for_SSL-TLS_(OWASP-CM-001).

[30] P. Eckersley and J. Burns. An observatory for the SSLiverse, July 2010. Talk at DEFCON '18. Available at: https://www.eff.org/files/DefconSSLiverse.pdf.

[31] P. Eckersley and J. Burns. Is the SSLiverse a safe place, 2010. Talk at 27C3. Available at: https://www.eff.org/files/ccc2010.pdf.

[32] P. Mockapetris. Domain Names - Implementation and Specification, 2008. RFC1035.

[33] Y. Pan and X. Ding. Anomaly Based Web Phishing Page Detection. In *Proceedings of the 22nd Annual Computer Security Applications Conference*, ACSAC '06, pages 381–392, Washington, DC, USA, 2006. IEEE Computer Society.

[34] S. Moriai, A. Kata, and M. Kanda. Addition of Camellia Cipher Suites to Transport Layer Security (TLS), July 2005. RFC4132.

[35] M. Schloesser, B. Gamble, J. Nickel, C. Guarnieri, and H. Moore. Project Sonar - IPv4 SSL Certificates. Available at: https://community.rapid7.com/community/infosec/sonar.

[36] SSL Labs. SSL Server Rating Guide. Technical report, SSL Labs, 2009. Available at: https://www.ssllabs.com/downloads/SSL_Server_Rating_Guide_2009d.pdf.

[37] The CA/Browser Forum. Guidelines For The Issuance And Management Of Extended Validation Certificates. Available at: https://cabforum.org/wp-content/uploads/EV-Code-Signing-v.1.2.pdf.

[38] Vasco. Diginotar reports security incident. August 2011. Available at: https://www.vasco.com/company/about_vasco/press_room/news_archive/2011/news_diginotar_reports_security_incident.aspx.

[39] N. Vratonjic, J. Freudiger, V. Bindschaedler, and J.-P. Hubaux. The Inconvenient Truth About Web Certificates. In B. Schneier, editor, *Economics of Information Security and Privacy III*, pages 79–117. Springer New York, 2013.

[40] X. Wang and H. Yu. How to Break MD5 and Other Hash Functions. In *Proceedings of the 24th annual international conference on Theory and Applications of Cryptographic Techniques*. Springer-Verlag, May 2005.