The SSL Landscape – A Thorough Analysis of the X.509 PKI Using Active and Passive Measurements

Ralph Holz, Lothar Braun, Nils Kammenhuber, Georg Carle
Technische Universität München
Faculty of Informatics
Chair for Network Architectures and Services
{holz,braun,kammenhuber,carle}@net.in.tum.de

ABSTRACT

The SSL and TLS infrastructure used in important protocols like HTTPs and IMAPs is built on an X.509 public key infrastructure (PKI). X.509 certificates are thus used to authenticate services like online banking, shopping, e-mail, etc. However, it always has been felt that the certification processes of this PKI may not be conducted with enough rigor, resulting in a deployment where many certificates do not meet the requirements of a secure PKI.

This paper presents a comprehensive analysis of X.509 certificates in the wild. To shed more light on the state of the deployed and actually used X.509 PKI, we obtained and evaluated data from many different sources. We conducted HTTPs scans of a large number of popular HTTPs servers over a 1.5-year time span, including scans from nine locations distributed over the globe. To compare certification properties of highly ranked hosts with the global picture, we included a third-party scan of the entire IPv4 space in our analyses. Furthermore, we monitored live SSL/TLS traffic on a 10 Gbps uplink of a large research network. This allows us to compare the properties of the deployed PKI with the part of the PKI that is being actively accessed by users.

Our analyses reveal that the quality of certification lacks in stringency, due to a number of reasons among which invalid certification chains and certificate subjects give the most cause for concern. Similar concerns can be raised for other properties of certification chains and also for many self-signed certificates used in the deployed X.509 PKI. Our findings confirm what has long been believed – namely that the X.509 PKI that we use so often in our everyday's lives is in a sorry state.

Categories and Subject Descriptors

C.2.2 [**Computer-Communication Networks**]: Network Protocols—*SSL, TLS, X.509*; C.2.3 [**Computer-Communication Networks**]: Network Operations—*SSL, TLS, X.509*; E.3 [**Data Encryption**]: Public key cryptosystems, Standards, SSL, TLS, X.509; H.4.3 [**Information Systems Applications**]: Communications Applications—*Electronic mail, Information browsers, SSL, TLS*

General Terms

Security, Measurement, Human Factors

Keywords

SSL, TLS, HTTPS, X.509, Certificates, Public Key Infrastructure

1. INTRODUCTION

Security, privacy and data integrity are important properties of today's Internet applications and protocols. As the Internet is being used for many commercial activities, such as shopping, online banking, or electronic trading, the value of data has increased. Networking protocols and applications are therefore increasingly expected to protect critical data by providing encryption, data integrity, and, most importantly, entity authentication. As these security goals are difficult to attain, application developers and protocol designers often rely on well-established security layers below their own protocols. The SSL and TLS protocol suites are used by many applications and protocols to provide a certain level of security, as they build on well-understood and thoroughly-analyzed cryptographic algorithms. Authentication is done using a public key infrastructure (PKI) built with the X.509 standard.

Cryptographic algorithms can be mathematically analyzed with respect to the security they offer, and implementations of these algorithms can be checked for correctness. Hence, it is possible to estimate the level of security they provide with a fair amount of confidence. In contrast, X.509 infrastructures depend not only on cryptography, but also on various organizations and entities that are required to conduct their work in conformance with abstract process descriptions. Certification Authorities (CAs), for example, certify the identities and public keys of other entities in an X.509 certificate. CAs need to conduct an identity check at a certain level of thoroughness before issuing a certificate. However, these CA-internal work processes are difficult to assess from the outside, and thus users need to place their trust into the correct working of a CA. As many of these processes involve human work, factors such as negligence or malice can introduce problems – yet entity authentication relies completely on the correct execution of identification processes. Web clients are a good example for this: browser vendors decide which CAs are included in their so-called Root Stores, i.e., their lists of trusted CAs. This reveals a classic dilemma: although this method removes control from end-users, it is difficult to imagine a better method, as it would likely be too much to ask of users themselves to assess the trustworthiness of CAs. However, the number of trusted CAs has grown very large (more than 150 in the case of Mozilla Firefox). This raises the question: can we assess the quality of the PKI that has thus been established? Although we cannot verify the implementation of work processes

within a CA from outside, we can observe and analyze the results of these processes: the deployed X.509 public key infrastructure.

This paper conducts a thorough analysis of the currently deployed and practically used X.509 infrastructure for TLS/SSL and examines its security-related properties. To obtain an overall picture of the deployed PKI, we collected X.509 certificates over a time-span of more than 1.5 years, and from several measurement points. Furthermore, we used passive traffic measurements on a 10Gbps link to obtain a picture of which parts of the PKI are actually used. We observed about 250 million TLS/SSL sessions over a four-week time span and extracted certificates from these sessions. We then evaluated the security-relevant properties of all acquired certificates. Last, but not least, we included data from previous third-party work into our analysis to be able to extend our view and compare our work with previous evaluations.

Contributions. Using these data sets, we evaluated the state of the currently deployed infrastructure and estimated its quality as encountered by users. We can show how often popular hosts offer TLS/SSL and what the negotiated ciphers and key lengths are. Most importantly, we present a number of results that show the certification infrastructure to be broken at several points: certification chains are often invalid, host names in subjects are frequently incorrect, and many certificates are re-used on too many hosts. This leads to a very low number of certificates that are accepted by e.g., a Web browser: only one out of five certificates can be counted as absolutely valid. Even then, some of these certificates exhibit weaknesses like weak signature algorithms. We also show which errors in certification chains are most common, how long these chains are, and the surprisingly small number of distinct chains that are actually used. Furthermore, we assess the properties of public keys and signature algorithms. We also include an analysis of occurrences of cryptographically weak keys. Thanks to the long-time observations and the geographic distribution, we are able to capture and describe the rather slow development of the X.509 PKI.

Organization. The remainder of this paper is organized as follows: Section 2 introduces X.509 and the structure of the PKI that is built upon the standard. We highlight relevant security parameters and properties, possible flaws in the processes that are responsible for setting these parameters, and the security implications of such errors. Section 3 presents related work that previously analyzed the X.509 infrastructure, and discusses how our work differs and extends these previous evaluations. Our data sets, their properties and the active and passive measurement methodology that we used to obtain the data are presented in Section 4. Section 5 presents the actual analysis of security-related PKI properties based on these data sets. Our paper concludes with a discussion in Section 6.

2. X.509 PUBLIC KEY INFRASTRUCTURE

This section introduces the public key infrastructure that is used by SSL and TLS. We describe those parts of X.509 that are relevant for this paper.

X.509 is an ITU-T standard for a public key infrastructure (PKI) and has been adopted by the Internet Engineering Task Force (IETF) as the PKI for several IETF protocols [1]. Thus, X.509 certificates are an integral part of the SSL and TLS protocol suites [2]: most often, they are used for server authentication in TLS/SSL, where the server presents its certificates to the client. Certificates are thus important for various protocols such as HTTPs, IMAPs, SMTPs and POP3s.

X.509 defines a somewhat complex certification infrastructure. Certification Authorities (CAs) are issuers of certificates, which are essentially a cryptographic binding of identity and public key. The binding is achieved with a digital signature. The identity is stored

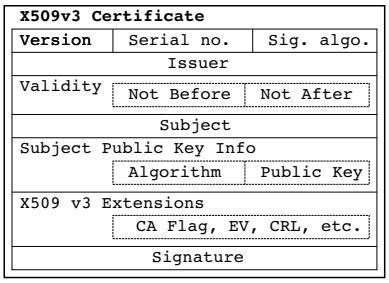

Figure 1: Schematic view of an X.509v3 certificate.

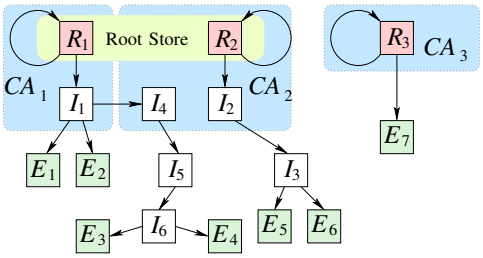

Figure 2: X.509 certificate chain examples.

in the so-called *subject* field of the certificate (Fig. 1). Every CA must ensure it only issues a certificate to the correct entity that is identified in the subject. This is most commonly a DNS host name.

As certificates contain information about communication peers, they are commonly exchanged at the beginning of a session setup. A communication peer must analyze its peer's certificate in order to determine whether it considers it to be the desired communication partner. To this end, it checks whether the certificate was issued and signed by a CA that it has trust in. Then, it checks if the information in the certificate identifies the other peer as the intended communication partner. In the example of the WWW, a browser needs to check if the domain in the certificate really is the domain it intends to connect to. Apart from identity and public key information, a certificate contains further important data, e.g., a validity period.

The X.509 PKI can be viewed as a tree or, since many CAs exist, a forest. Figure 2 displays a toy example for such a forest. CAs reside at the top and issue certificates. For this purpose, they issue certificates to themselves (*self-signed certificates*), so-called *Root Certificates* (R_x). These are then used for signing, i.e., issuing certificates. Instead of issuing certificates directly to end-entities (E_x), it is allowable – and in practice often done – that CAs issue intermediate certificates (I_x) that are then used for further signing processes. There are several reasons for this. First, it allows CAs to delegate the identification process to other authorities. This is especially useful for globally operating CAs as it spreads the workload to local registrars, who are supposedly also more competent at local identification procedures. However, intermediate certificates do not necessarily identify other authorities. They can also be used for security reasons: they allow a CA to sign an intermediate certificate with their Root Certificate. The intermediate certificate can then be used in the CA's online issuing business and can be replaced or revoked at any time, whereas the Root Certificate (R_1, R_2) is kept safe off-line. An intermediate certificate may be used to sign further intermediate certificates. The result is a so-called *trust chain* or *certification chain*, which may (in theory) be arbitrarily long.

The number of intermediate authorities and certificates thus increases the number of points for attack. It also removes some con-

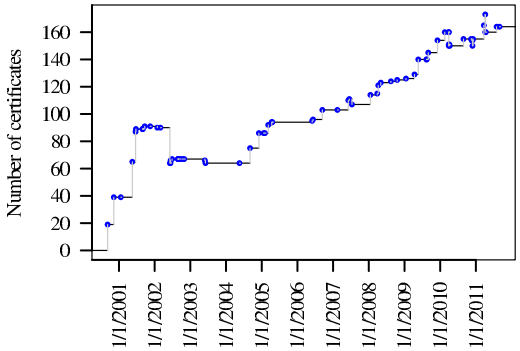

Figure 3: Growth of the NSS/Mozilla Root Store.

trol from the client – it is debatable if users would trust all intermediate authorities even if they knew about them. To understand the extent of the problem, it is important to realize that *any* CA can issue a certificate for *any* domain name.

For certificate checking to work, Root Certificates have to be known to the verifying party. This is solved in Web browsers by shipping a so-called *Root Store*, which contains the Root Certificates of selected CAs. In our example, R_1 and R_2 are in the Root Store while R_3 is not; hence the browser would reject E_7 as untrusted and display a warning message. An interesting case is *cross-signing*, where a CA also has a certificate (I_4) that has been actually signed by a Root Certificate of another CA. This is useful if the other CA is contained in a Root Store in which the cross-signed CA is not contained. However, Root Stores have a very undesired effect. If a CA or any of the subordinate intermediates it cooperates with becomes compromised, the attacker can issue certificates for *any* domain. It has been documented that this has happened several times [3, 4] in 2011: an attacker (purportedly the same) hacked several CAs such that he was able to issue certificates for several high-value domains to himself. In the first case [3], the number of issued certificates was low, and browser vendors responded by blacklisting the forged certificates. In the latter case [4], the number of forged certificates exceeded 500, and there were claims that the certificates had been used on a state level in a Man-in-the-middle attack. This led to the removal of the affected Root CA from the Root Store, an unprecedented step in the history of the X.509 PKI for the WWW.

CAs are included in the Root Store by vendor-specific processes. In the case of Mozilla, this is an open forum; but this is not the case for large companies like Microsoft or Apple. It should be noted that the number of Root CAs in browsers has been growing for a long time. We verified this for Mozilla Firefox, whose Root Store comes with the NSS crypto library that it uses. Figure 3 shows the development of the NSS Root Store since the year 2000. Starting out with about only a handful of certificates, the number has greatly increased over the years. Until December 2010, the number has grown to 160, the latest Root Store we used for this paper.

In order to be valid, a certificate must fulfill several properties, which we will present in detail later. These properties have different security implications for the overall security of the TLS/SSL connections. Also, it should be noted that users are today very used to browser warnings when certificates or trust chains are found to be broken. Studies like [5] showed that, for various reasons, the typical reaction is to ignore warnings and continue with the connection. All in all it is quite correct to conclude that the X.509 PKI is a very fragile construction. In fact, criticism has been voiced by prominent researchers [6, 7].

3. RELATED WORK

We are aware of two previous contributions on certificate analysis for TLS/SSL. Both were given as talks at hacking symposia, but have not been published as articles. Between April and July 2010, members of the Electronic Frontier Foundation (EFF) and iSEC Partners conducted a scan of the entire IPv4 space on port 443 and downloaded the X.509 certificates. Initial results were presented at [8] and [9]. A second scan was conducted in August 2010. The authors focused on determining the certification structure, i.e., number and role of CAs, and several noteworthy certificate properties like "weird" subjects (e.g., `localhost`) or issuers.

Ristic conducted a similar scan like EFF in July 2010 and presented some results in talks at BlackHat 2010 [10] and again (now including the EFF data) at InfoSec 2011 [11]. The initial scan was conducted on 119 million domain names, and additionally on the hosts on the Alexa Top 1 Million list [12]. He arrived at about 870,000 servers to assess, although the exact methodology cannot be derived from [10, 11]. However, the information about certificates and ciphers collected is the same as in our scans, and together with the EFF data set, our data sets provide a more complete coverage.

Lee et al. also conducted a scan of TLS/SSL servers [13]. In contrast to our work, they did not investigate certificates but focused on properties of TLS/SSL connections (i.e., ciphers, MACs, etc.) and the cryptographic mechanisms supported by servers. The number of investigated servers was much lower (20,000).

Yilek et al. investigated the consequences of the Debian OpenSSL bug of 2008 [14]. A bug in the OpenSSL implementation had caused very weak randomness in key generation, and it was possible to pre-compute the affected public/private key combinations. The authors traced the effect of the error over a time of about 200 days and scanned about 50,000 hosts.

The problem with scans is generally that hosts are included that are not intended to be accessed with TLS/SSL and thus provide invalid (and often default) certificates. The percentages given in [8, 9, 10, 11] thus need to be treated with caution. Our actively obtained data sets concentrate on high-ranked domains from the Alexa Top 1 Million list, and observe these domains over a long time period. Note that high-ranked domains can be assumed to be more aimed at use with TLS/SSL. This should at least be true for the top 1,000 or top 10,000. Our monitoring does not suffer significantly from this problem. Thanks to it, we are not only able to estimate the deployment of the TLS/SSL infrastructure, but to analyze which parts of the PKI are actively used and therefore seen by users. Furthermore, our view on the TLS/SSL deployment is not a single snapshot at an arbitrary time, but includes changes that operators have conducted in 1.5 years. Moreover, by analyzing TLS/SSL data that has been obtained from all over the world, we can even estimate how users see the TLS/SSL-secured infrastructure in other parts of the world. Finally, we include the EFF data from the mentioned related work into our own evaluation, and are therefore able to enhance previous results by applying our own evaluation algorithms, providing more comparability between our work and theirs.

4. DATA SETS

This section presents the various data sets that we obtained for our analyses. Section 4.1 presents our approach and the tools we used to perform our active TLS/SSL scans. Our setup and tool chain for our passive measurement of TLS/SSL traffic is explained in Section 4.2. All data sets that we use in our analyses, including the data sets from related work, are described in Section 4.3. Table 1 gives an overview of all data sets for this work. The data sets

Short Name	*Location*	*Time (run)*	*Type*	*Certificates (distinct)*
Tue-Nov2009	Tübingen, DE	November 2009	Active scan	833,661 (206,588)
Tue-Dec2009	Tübingen, DE	December 2009	Active scan	819,488 (205,700)
Tue-Jan2010	Tübingen, DE	January 2010	Active scan	816,517 (204,216)
Tue-Apr2010	Tübingen, DE	April 2010	Active scan	816,605 (208,490)
TUM-Sep2010	Munich, DE	September 2010	Active scan	829,232 (210,697)
TUM-Nov2010	Munich, DE	November 2010	Active scan	827,366 (212,569)
TUM-Apr2011	Munich, DE	April 2011	Active scan	829,707 (213,795)
TUM-Apr2011-SNI	Munich, DE	April 2011	Active scan with SNI	826,098 (212,229)
Shanghai	Shanghai, CN	April 2011	Active scan	798,976 (211,135)
Bejing	Beijing, CN	April 2011	Active scan	797,046 (211,007)
Melbourne	Melbourne, AU	April 2011	Active scan	833,571 (212,680)
İzmir	İzmir, TR	April 2011	Active scan	825,555 (211,617)
São Paulo	São Paulo, BR	April 2011	Active scan	833,246 (212,698)
Moscow	Moscow, RU	April 2011	Active scan	830,765 (213,079)
Santa Barbara	Santa Barbara, USA	April 2011	Active scan	834,173 (212,749)
Boston	Boston, USA	April 2011	Active scan	834,054 (212,805)
MON1	Munich, DE	September 2010	Passive monitoring	183,208 (163,072), Grid: 47% (51.94%)
MON2	Munich, DE	April 2011	Passive monitoring	989,040 (102,329), Grid: 5.72% (24.40%)
EFF	EFF servers	March–June 2010	Active IPv4 scan	11,349,678 (5,529,056)

Table 1: Data sets used in this work.

that we acquired by active scans ourselves will be released to the scientific community at [15].

4.1 Active Scans

As a base for our scans, we used the Alexa Top 1 Million Hosts list [12] that was current at the time of the respective scan. This list contains the most popular hosts of the WWW as determined by the ranking algorithm of Alexa Internet, Inc. These are thus sites that many users are likely to visit. Although its level of accuracy is disputed [16], it nevertheless is appropriate for our need to identify popular hosts. Since the list's format is somewhat inconsistent, we emulated browser behavior by expanding each entry to two host names: one with `www` prefix and one without.

Every scan was preceded by `nmap` scans on TCP port 443 to filter out hosts where this port was closed. The actual certificate scans thus started two weeks after obtaining the Alexa list. Our SSL scanning tool itself uses OpenSSL. It takes a list of host names as input and attempts to conduct a full TLS/SSL handshake on port 443 with each of them. Where successful, the full certificate chain as sent by the server is stored, along with further data, such as TLS/SSL connection properties.

4.2 Passive Monitoring

We monitored all TLS/SSL traffic entering and leaving the *Munich Scientific Research Network* (MWN) in Munich, Germany. The network interconnects three major universities as well as affiliated research institutions in the greater Munich area and provides Internet access to their users via a 10 Gbit/s link to the German research network DFN. It serves about 120,000 users with approximately 80,000 devices. During busy hours, the average link load amounts to approximately 2 Gbit/s inbound and 1 Gbit/s outbound traffic (as of April 2011). We obtained passive monitoring data in two runs. One problem we had to cope with was the high link speed. We improved our monitoring software setup between the first and second run, whereas the hardware (a four-core Intel Core i7 with hyper-threading using an Intel 10 GE network interface with 82598EB chipset) remained the same.

In order to deal with the huge amount of traffic, both runs used a sampling algorithm that samples the first n bytes of each bi-flow [17]. This is sufficient, as the handshake messages including the X.509 certificates are exchanged at the beginning of a TLS/SSL session setup.

For our first monitoring run, we used a process similar to the one used by the *Time Machine* [18]: we captured the beginning of every observed bi-flow and dumped all sampled packets to disk, starting a new file as soon as a dump file reached 10 GB. Whenever a dump file was finished, the TLS/SSL connections were extracted offline. Due to disk I/O and disk space limitations, we were only able to sample the first 15 kB of each bi-flow.

The second monitoring run in April 2011 was conducted using online TLS/SSL analysis. We used an optimized capturing setup, including our improvement presented in [19], based on TNAPI [20] to build a multi-core aware online analysis system. We now were able to run six instances of our monitoring application in parallel. Each instance employed a sampling process that sampled the first n kB of each bi-flow. Using this parallelization technique, we were able to analyze for each bi-flow up to 400 kB of traffic data while suffering less than 0.003% overall packet loss.

As the TLS/SSL processing tool, we used the intrusion detection and protocol parsing system Bro [21] in both monitoring runs. Using Bro's dynamic protocol detection feature [22], we were able to identify TLS/SSL in a port-independent way. We used Bro-1.5 with some applied patches to the Bro code, which fix some issues and allow us to extract and store complete certificate chains from the monitored connections.

4.3 Data Properties

Table 1 summarizes the locations, dates and number of certificates in the different sets. Table 2 provides additional details for the monitoring runs. Our data sets can be grouped into four classes.

First, we conducted scans from hosts that were located in Germany at the University of Tübingen and at TU München. These scans were carried out between November 2009 and April 2011, thus spanning a time interval of about 1.5 years. In April 2011, we performed an extra scan with SNI enabled. SNI is a TLS extension to address virtual HTTP hosts. With SNI, the host name is passed in the TLS handshake so the server can select an appropriate certificate to deliver (note that the TLS/SSL handshake takes place *before* the client sends the HTTP header).

The second group of data sets was obtained in April 2011. We employed PlanetLab [23] nodes from different countries, in order to obtain a geographically distributed picture of the TLS/SSL deployment. We wanted to determine whether location-dependent factors like content distribution networks (CDNs), which use DNS to route

clients to different computing centers depending on the geographic location of the client, would also result in different certificates. This can yield information on the state and practice of certificate deployment in CDNs. Furthermore, we had the hope that, as a side effect, we might be able to determine whether certain locations actively interfere with TLS/SSL traffic by swapping end-host certificates during connection establishment (Man-in-the-middle).

The third group of data sets was obtained by passively monitoring live TLS/SSL traffic. The important distinction between certificates obtained from passive monitoring and those obtained by scans is that these certificates reflect that part of the PKI infrastructure that is not only deployed but also actually used due to user access to sites. We extracted all observed certificates in TLS/SSL traffic over each two-week period. In September 2010, we were able to observe in total more than 108 million TLS/SSL associations, resulting in over 180,000 certificates, of which about 160,000 were distinct. Our second run observed, during a similar time span of two weeks, more than 140 million TLS/SSL connections, which were responsible for about 990,000 certificates, of which about 100,000 were distinct.

As the MWN is a research network that hosts several high-performance computing clusters, we observed much Grid-related traffic secured by TLS/SSL. Although most TLS/SSL connections are due to HTTPs, IMAPs and POPs (see Table 2 for details), we encountered a rather large number of Grid certificates, as Grid certificates were often replaced (their median validity period was just 11 hours). As Grid-related certificates are not comparable to those used on the WWW, we filtered them out in our analysis of certificate properties. We explain this in Section 4.4.

Finally, we included a data set from related work into our analysis. This data set was obtained using a different scanning strategy. The EFF data set is based on scanning the entire assigned IPv4 address space, which took a few months. This results in a higher number of observed certificates, but does not provide a mapping onto DNS names. Hence, the data set cannot be verified in terms of a matching subject for a host name, since information about which domain was supposed to be served with a given certificate is not contained. In contrast, our own data sets provide this information.

4.4 Data Pre-Processing

The International Grid Trust Federation [24] operates an X.509 PKI that is separate from the one for HTTPs and other TLS/SSL-related protocols. Their Root Certificates are not included in the Firefox Root Store, but are distributed in Grid setups.

Our setup stored certificates without a reference to the (potentially many) TLS/SSL connections during which it was observed. Although we cannot filter out Grid traffic in our analysis of TLS/SSL connections to hosts (Section 5.1), we were still able to identify some properties of Grid traffic by correlating the encountered IP addresses with those of known scientific centers.

For our analysis of certificate properties (Section 5.2), however, it is possible to identify Grid-related certificates and filter them out. Our filter mechanism is rather coarse: we simply check whether or not a certificate contains the word 'Grid' in the issuer field. We acknowledge this is imperfect, but it is much less expensive to conduct than verifying if a certificate chains up to one of the Grid Root CAs. We tested our certificate filter by checking the certification chains in the sub-data set of Grid certificates. Indeed, 99.95% of them could not be found to chain to a CA in the Firefox Root Store, and not one of them had a valid certification chain (as is to be expected due to the use of a different PKI setup). At the same time, the values for validity of certification chains of non-Grid certificates was in the same range as with our active scans.

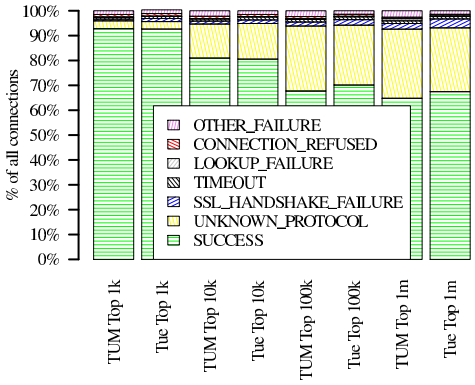

Figure 4: TLS/SSL connection errors for Tue-Nov2009 and TUM-Apr2011 scans, in relation to Alexa rank.

We thus conclude that our filter has removed most of the Grid certificates and the remaining bias is tolerable.

5. RESULTS OF OUR INVESTIGATIONS

We now describe results from our analyses, going item-wise through each question that we address. For each item, we explain its relevance for the security of the TLS/SSL infrastructure.

An important distinction that we sometimes make is whether we analyzed certificates of the full set or just the *distinct* certificates in the data set. The difference is that the former refers to the deployment as we have seen it and includes duplicate certificates, i.e., certificates that are used on more than one host. The latter is a pure view at certificates as they have been issued.

5.1 Host Analyses

Before investigating actual certificates, we first present some properties of the TLS/SSL hosts that we either contacted through active scanning, or whose connections we monitored passively.

Host Replies with TLS/SSL. A very central question is how many hosts actually support TLS/SSL and allow TLS/SSL connections on ports that are assigned to these protocols. To determine how many hosts offer TLS/SSL on the HTTPs port, we evaluated replies to our OpenSSL scanner described in Section 4.1. Figure 4 shows results for the scans in November 2009 and April 2011. It gives an overview on all queried hosts, the top 1,000, the top 10,000, and the top 100,000. As can be seen, the numbers do not change significantly over time, neither for the overall picture nor for the top 1,000. Two thirds of all queried hosts offer TLS/SSL on port 443, and more than 90% of the top 1,000 do so, too.

The number of cases where an unknown protocol was used on the HTTPs port is surprising, however. We therefore re-scanned all hosts that had previously shown this result and analyzed samples of the re-scan traffic manually. In all samples, the servers in question offered plain HTTP on the TLS/SSL port. As can be seen, this unconventional configuration is less popular with highly ranked servers.

Handshake failures during TLS/SSL setup can also result in failure of the association setup. However, the overall number of handshake failures and the number of other (i.e., not grouped otherwise) failures is rather low.

All in all, only about 800,000 hosts from the expanded Alexa list allowed successful TLS/SSL handshakes on the HTTPs port. Since a number of important sites (Google, Facebook, Twitter, ...)

Property	*MON1*	*MON2*
Connection attempts	108,890,868	140,615,428
TLS/SSL server IPs	196,813	351,562
TLS/SSL client IPs	950,142	1,397,930
Different server ports	28,662	30,866
Server ports $\leq$ 1,024	91.26%	95.43%
HTTPs (port 443)	84.92%	89.80%
IMAPs and POPs (ports 993 and 995)	6.17%	5.50%

Table 2: TLS/SSL connections in recorded traces.

have switched to offer TLS/SSL by default, we find it surprising that there does not seem to be a marked change in the overall deployment in the past 1.5 years either.

Our passive monitoring data shows quite a large number of TLS/SSL-enabled servers. This is related to Grid traffic, which we did not filter out for this analysis. The numbers changed markedly between the monitoring intervals in September 2010 and April 2011, although both runs were conducted over a period of two weeks and both time intervals were during university semester breaks. We found differences in the number of observed TLS/SSL connections, and in the number of clients and servers that were involved in this communication. These differences are summarized in Table 2. The number of connections between the first and second monitoring runs increased from 108 million to 140 million connections. Both the number of TLS/SSL servers and the number of the observed TLS/SSL clients have increased. The increase in observed TLS/SSL servers is also the main factor responsible for the increase in the observed certificate numbers in Table 1. The vast majority of observed TLS/SSL traffic used well-known ports ($< 1,024$), especially HTTPs, IMAPs and POPs. We can show that the remaining traffic was Grid-related on several occasions as the involved IP addresses are assigned to scientific computing centers.

Negotiated Ciphers and Key Lengths. The strength of the cryptographic algorithms and the length of the involved symmetric keys determine the cryptographic protection of the TLS/SSL connection. From our monitoring data, we obtained statistics on the ciphers and key lengths used to encrypt TLS/SSL connections. Data from active scans was not involved: it has only limited validity here, as the negotiated ciphers highly depend on what a client supports; hence, all results from active scans would be biased.

Figure 5 presents the most frequently negotiated ciphers, key lengths and digests encountered in the monitoring runs. Generally, strong ciphers (AES, Camellia, RC4) were selected with appropriate key lengths, sometimes offering a very good safety margin (256 bit). 3DES is still used, but not in the majority of cases. The use of MD5 for Message Authentication Codes, although not problematic at this time, is not encouraged [2]. However, it is still extensively used: the most popular cipher/digest combination (`RSA_WITH_RC4_128_MD5`) in September 2010 employed MD5 hashes for digest calculation. Fortunately, we can see that the popularity of this combination is decreasing: while it is still the most popular combination in April 2011, we can see an increasing share of SHA-based digest algorithms.

When comparing our results to those from 2007 in [13], we find that while the two most popular algorithms remain AES and RC4, their order has shifted. In 2007, AES-256, RC4-128 and 3DES were the default algorithms chosen by servers, in that order. In our data, we find the order is RC4-128, AES-128 and AES-256. It is difficult to give a compelling reason here. It could be that more clients support TLS/SSL now and their cipher support is different; but it could also be that more servers support TLS/SSL now and

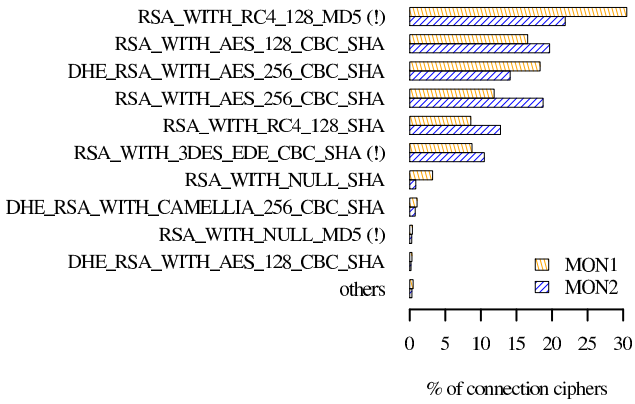

Figure 5: Top 10 chosen ciphers in passive monitoring data.

their default choice is the very fast RC4 at lower key length. If so, this is a debatable choice: although RC4 is secure, great care must be taken to avoid certain attack vectors [25].

Furthermore, we can see that some of the connections, albeit a minority, chose no encryption for their user traffic during the handshake. Such `NULL` ciphers were observed for 3.5% of all connections in MON1, and in about 1% of all connections in MON2. Again, we can show that the corresponding IP addresses are assigned to computing centers, and the connections are likely Grid traffic. Our hypothesis is that TLS/SSL is only used for authentication purposes, whereas encryption is omitted due to performance reasons.

5.2 Certificate Properties

We investigated several security properties of the X.509 PKI that are related to properties of certificates and certification chains. In this analysis, we filtered out Grid-related certificates in the data we had obtained from passive monitoring.

Certificate Occurrences. Ideally, every host should have its own certificate to allow TLS/SSL connections. Since the SNI extension does not enjoy real deployment, however, it is quite common that a certificate is issued for several domain names. This is not the only reason: it may be less costly to buy a certificate from a CA that includes several host names than to buy certificates for each host name. It may also be less time-consuming for the operator. However, as the private key for every certificate must also be stored with the public key, this can potentially increase the attack surface if a certificate is used on more than one physical machine.

We thus checked how often the same certificate is reused on several hosts. Figure 6 shows the complementary cumulative distribution function (CCDF) of how often the same certificate was returned for multiple different host names during the active scan of September 2010, in the middle of our observation period. We can see that the same certificate can be used by a large number of hosts (10,000 or more), although the probability for this is significantly less than 1:10,000. However, the probability that a certificate is reused rises quickly for a smaller number of reuses. It is not uncommon (about 1% of the cases) that 10 hosts share the same certificate.

We thus investigated which hosts are particularly prone to reuse a certificate. Hosts are identified by the domain names in the certificate's subject field. Figure 7 shows these for the certificates that we found most often. Most of these are identifiable as Web hosters – but only the certificates for `*.wordpress.com` were actually valid

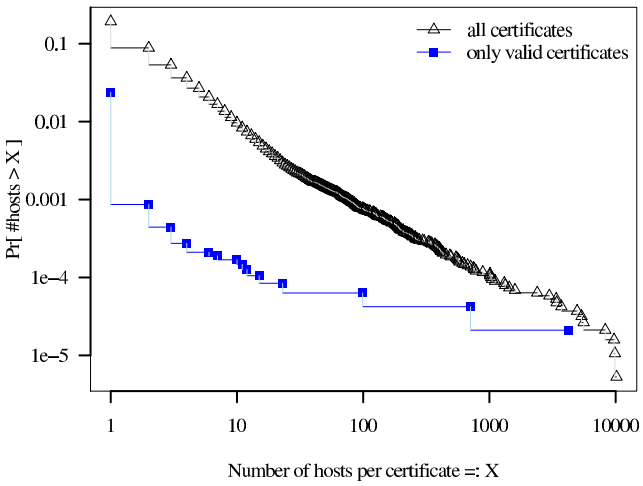

Figure 6: Certificate occurrences: CCDF for number of hosts per distinct certificate.

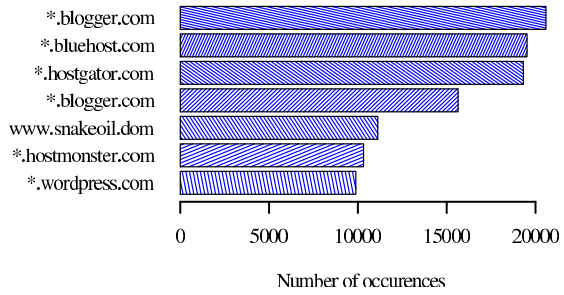

Figure 7: Domain names appearing in subjects of certificates most frequently used on many hosts.

(correct chain, correct host name). This is a rather poor finding, considering how popular some of these hosters are.

`www.snakeoil.dom` is a standard certificate for the Apache Web server, possibly appearing here because of Apache servers that run a default configuration (but probably are not intended to be accessed via HTTPs). We continued to investigate host names for those certificates that occurred at least a 1,000 times, and found that these seemed to be primarily Web hosting companies. We explore the issues of domain names in the subject fields and correctness of certificates in the sections below.

Validity of Certification Chains. There are several factors that determine whether a browser should accept a certificate as correct or not. The first one is whether the certification chain presented by a server is correct, i.e., the chain is complete and leads to a Root Certificate in the browser's Root Store, no certificate in the chain has expired and all signatures in the chain verify, etc.

There are a number of causes for a broken chain. We thus investigated the correctness of chains with respect to the Firefox Root Store from the official developer repositories at the time of the scan or monitoring run. Note that a missing Root Certificate only means that the certificate is not valid in a default Firefox, but the Root Store of a client can be reconfigured to contain the required Root Certificate. Moreover, some CAs have not yet been included in Root Stores, although a number of people consider them trustworthy. A prominent example is CACert.org, a community-driven nonprofit CA.

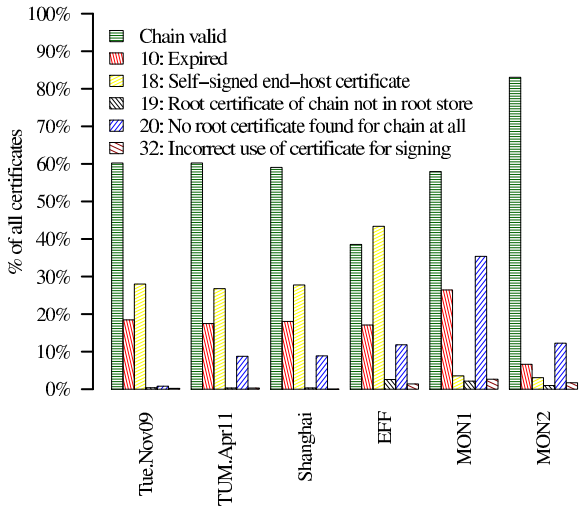

Figure 8: Error codes in chain verification for various data sets. Multiple errors can cause the sums to add up to more than 100%.

We used OpenSSL's `verify` command to check certification chains. Our check verified the full certification chain. This step does not check whether the subject in the certificate has a certain value (e.g., the correct host name) – this has to be checked separately. We examined which errors occurred how often in the validation of a certification chain (N.B.: a single chain can report multiple errors). Figure 8 presents our results for a selection of our data sets. The error codes are derived from OpenSSL error codes. In each case, a browser would display a warning to the user (usually with the option to override the alert). Depending on the vendor, additional user clicks are necessary to determine the exact cause. Error code 10 indicates the end-host certificate was expired at the time we obtained it. For the monitoring runs, we allowed a grace period here until the end of the run, as we had not logged timestamps due to disk space limitations. Expired certificates can be considered in two ways: either as completely untrustworthy, or just as less trustworthy than certificates within their validity period. The latter would account for human error, i.e., failure to obtain a new certificate. Error code 18 identifies an end-host certificate as self-signed. This means no certification chain at all is given, a user has to trust the presented certificate and can only determine its validity out-of-band (e.g., by manually checking hash values) [1]. Error codes 19–21 indicate a broken certification chain. Error code 19 means a correct full chain was received, but the Root Certificate is not in the Root Store and thus untrusted. This error can occur, for example, if a Web site chooses to use a Root Certificate that has not been included in a major browser. As explained, certain organizations (like some universities) sometimes use Root CAs of their own and expect their users to manually add these Root Certificates to their browsers. If this is not done securely out-of-band, its value is very debatable – we have anecdotal experience of university Web sites asking their users to 'ignore the following security warning', which would then lead them to the 'secured' page. Error code 20 is similar: there was a certificate in the sent chain for which no issuing certificate can be found, neither in the sent chain nor in the Root Store. Error code 21 means that only one certificate was sent

[1]Certificate pinning has been suggested as useful here. The idea is to imitate the trust model of SSH where a host key is viewed as correct on first encounter and its key remembered by the client for verification upon revisit.

(i.e., no chain) and there is no Root Certificate in the Root Store that would match the issuer of the end-host certificate. Error code 32 is interesting: it means that one or more certificates in the chain are marked as not to be used for issuing other certificates.

Some certificates found in all scans were found to have broken or nonsensical validity periods. Since there are less than 10 occurrences in any scan, we did not investigate them further. We also found cases where signatures are wrong and cannot be verified. Ratios were between 0.01% and 0.1%.

Figure 8 reveals that trust chains were valid in about 60% of cases when considering the active scans and over all certificates. Expired certificates (about 18%) and self-signed certificates (about 25%) are by far the most frequent errors found. The number of valid certificates does not change significantly when considering distinct certificates only. This means that issuing *and* deployment of certificates with valid trust chains are in line, which is a positive finding. Between November 2009 and April 2011, however, these numbers also remained constant. The same can be said for expired certificates: their number also remained around 18%. Note that the entries on the Alexa list had greatly differed by then; more than 550,000 hosts in April 2011 had not been on our expanded list in November 2009. We can thus say that even while the popularity of Web sites may change (and new sites likely enter the stage), the picture of this PKI with respect to valid certificates remains the same. This is a rather poor finding, as no improvement seems to occur.

Error code 20 increased in frequency: the number rose to around 8% in April 2011. We do not know a reason. Error codes 19 and 32 occurred very rarely. For our scans from Germany, we can thus conclude expired certificates and self-signed certificates are the major cause for chain verification failures. We compared our findings to the vantage points in Santa Barbara and Shanghai (Fig. 8; only Shanghai data shown) and found the perspective from these vantage points is almost the same.

Figure 8 also shows the findings for MON1 and MON2. Here, the situation is somewhat different. Although the number of valid chains was similar in MON1 (57.94%), it was much higher in MON2 (83.07%). We can also see that the number of errors due to expired certificates or with Error 20 has decreased. This does not mirror the findings from our scans. We did not find an immediate explanation for this phenomenon and will monitor it further.

We consequently also compared our results with the full data set of the EFF, which represents the global view for the IPv4 space. We found differences. First of all, the ratio of self-signed certificates is much higher in the EFF data set. This is not surprising, given that certification costs effort and often money – operators on not so high-ranking sites may opt for self-issued certificates, or just use the default certificates of common Web servers like Apache.

On the whole, however, the fact that about 40% of certificates in the top 1 million sites show broken chains is discouraging, even when considering that self-signed certificates are so common.

Correct Host Name in Certificate. The second major factor to determine whether a certificate should be accepted is the correct identification of the certified entity. The application itself needs to check if the subject of the certificate matches the entity it has contacted. The common practice on the WWW (but also for IMAPs and POPs) is that the client verifies that the subject certified in the certificate matches the DNS name of the server. In X.509 terminology and syntax, the subject field must be a *Distinguished Name* (DN). A *Common Name* (CN) is part of the DN. Very commonly, a DNS name is stored in the CN. Instead of writing the DNS name into the subject field, an alternative is to put it in the certificate's *Subject Alternative Name* (SAN) field. This can actually be viewed as desirable, as this is the intended purpose for this field [1]. It is, however, the less common practice. We checked if the CN attribute in the certificate subject matched the server's host name. Then, we checked if the SAN matched. We call certificates with correct chains where CN or SAN match 'absolutely valid'. Where the CN or SAN fields were wild-carded, we interpreted them according to RFC 2818 [26], i.e., `*.domain.com` matches `a.domain.com` but not `a.b.domain.com`. One exception was to count a single `*` as not matching, in accordance with Firefox's behavior. Note that we can conduct this investigation only for data sets from our scans, as neither data from monitoring nor the EFF data contain an indication of the requested host name.

In TUM-Apr2011, we found that the CNs in only 119,648 of the 829,707 certificates matched the host name. When we allowed Subject Alternative Names, too, this number rose to 174,620. However, when we restrict our search to certificates that also had correct chains, the numbers are 101,238 (correct host name in CN) and 149,900 (correct host name in CN or in SAN). This corresponds to just 18.07% of all certificates. We checked whether the picture changed for the data set of April 2011 where we had SNI enabled. This was not the case. The number of certificates with both valid chains and correct host names remained at 149,451 (18.09%). We deduce from this that client-side lack of SNI support is not the problem. We also determined the numbers for Tue-Nov2009, Tue-Apr2010 and TUM-Sep2010: they are 14.88%, 15.88% and 16.88%, respectively. This indicates a weak but positive trend.

Our findings mean that only up to 18% of certificates can be counted as absolutely valid according to the rules implemented in popular browsers – in a scan of the top 1 million Web sites. More than 80% of the issued certificates lead to a browser warning – we are not surprised that so many people are used to security warnings. These are major shortcomings that need to be addressed. However, we also have to add a word of caution here: while a poor finding, it is likely that many of these hosts are actually not intended to be accessed via HTTPs and thus neglect this critical configuration. Normal users may therefore never encounter the misconfigured site, even in the case of very popular sites. Still, omitting support for TLS/SSL does not increase security, either.

Unusual Host Names in the Common Name. We encountered a few unusual host names. In 60,201 cases (TUM-Apr2011-SNI, i.e., SNI enabled), we found the string 'plesk' as a CN. Our hypothesis was that this is a standard certificate used by the Parallels/Plesk virtualization and Web hosting environment. We tested this by rescanning the hosts, hashing the HTTP reply (HTML) and creating a histogram of this that counted how often which answer occurred. Just 8 kinds of answers were responsible alone for 15,000 variants of a Plesk Panel site, stating "site/domain/host not configured" or the like. As there are standard passwords for the Plesk Panel sites readily available via simple Google search, we find it problematic that such entry points cannot be authenticated in so many cases. A further favorite of ours are certificates issued for `localhost`, which we found 38,784 times. Fortunately, neither certificates with 'plesk' nor `localhost` were ever found to have valid chains.

However, by far the most answers occurred on relatively few hosts (2–10). We sampled some of these and found that HTTP redirections to normal HTTP sites were common after the TLS/SSL session establishment.

Host Names in Self-Signed Certificates. Server operators may opt to issue a certificate to themselves and act as their own CA.

EV Status	*Tue-Nov2009*	*TUM-Sep2010*	*TUM-Apr2011*	*Shanghai*	*Santa Barbara*	*Moscow*	*İzmir*
Yes	1.40%	2.10%	2.50%	2.56%	2.49%	2.51%	2.50%
No	98.60%	97.90%	97.50%	97.44%	97.51%	97.49%	97.50%

Table 3: Deployment of EV certificates over time and from Germany; and the same for April 2011 from Shanghai, Santa Barbara, Moscow and İzmir.

Hence, no external, responsible Certification Authority exists. This saves the costs for certification, but requires users to accept the self-signed certificate and trust it. The value of self-signed certificates is debatable: some view them as useful in a Trust-On-First-Use security model, as successfully used in the popular SSH protocol; others view them as contrary to the goals of X.509. Our own view is that self-signed certificates can be useful for personally operated servers, or where it is safe to assume that a Man-in-the-middle attack in the first connection attempt is unlikely. In the data set with enabled SNI (TUM-Apr2011-SNI) we checked if the self-signed certificates have a subject that matches the host name. The result is sobering: 97.78% of CNs do not match. Subject Alternatives Names matched in 0.5% of cases, but were rarely used (1.65% of certificates). Interestingly, the top 3 account for more than 50% of the different CNs. 'plesk' occurred in 27.30% of certificates, `localhost` or `localhost.localdomain` in 25.39%. The remaining CNs in the top 10 were usually in the range of 0.5–3%. The bulk of CNs is made up of entries that do not occur more than 1–4 times. Our conclusion is that self-signed certificates are not maintained with respect to host name. This does not make them useless in the above mentioned security model, but it certainly makes them puzzling when encountered by the average user.

Extended Validation (EV). Technically, a user should not only verify that the domain in the CN or SAN matches, but that other information in the certificate correctly identifies the entity on an organizational level, e.g., that it is really the bank he intended to connect to – and not a similar-looking phishing domain. This is the purpose of so-called Extended Validation certificates, which have been introduced several years ago. EV certificates are meant to be issued under the relatively extensive regulations described by the CA/Browser-Forum [27]. An *object identifier* in the certificate identifies it as EV. A browser is meant to signal EV status to the user (e.g., via a green highlight in the address bar).

We analyzed how often EV certificates are used; Table 3 shows the results. One can see that there is a light trend towards more EV certificates. We inspected the top 10,000 hosts in TUM-Apr2011 more closely and found that the ratio of EV certificates was 8.93%. For the top 1,000 hosts it was 8.11%, and for the top 100, 8.33%. Surprisingly, for the top 50 it was 5.17%. We found two explanations for this. First, Google sites dominate the top 50 (almost half of all hosts), and Google does not use EV. Second, a number of well-known Web sites (e.g., Amazon and eBay) use different hosts to let users log in. These are not in the top 50, but use EV.

Our conclusion here is that EV certificates are not very widespread, even though they can be very useful for sensitive domains (one may count Google's services among those, but also any bank). Since they are commonly more expensive than standard certificates, which are just issued against a domain name, this is somewhat to be expected. Even then, this is a sad finding.

Length of Certification Chain. As explained in Section 2, probabilities for negligence and errors can be assumed to increase with a high number of intermediate certificates if these are not subject to the same control as within the Root CA, e.g., if they are used by Intermediate CAs outside the direct control of the Root CA. Furthermore, servers sometimes sent more certificates than needed to build a valid chain. This can potentially reduce performance.

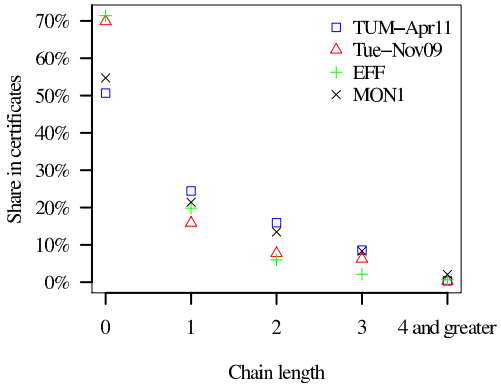

Figure 9: Chain characteristics for two scans, one monitoring run, and the EFF data. If no intermediate certificates are sent, the chain length is 0.

We thus computed the length of certification chains by only counting those certificates in the chain that were neither the end-host certificate, nor Root Certificates, nor self-signed intermediate certificates[2]. The latter cannot possibly contribute to an otherwise *valid* certification chain.

Figure 9 shows the result for two scans and one monitoring run from Germany, and a comparison with the EFF data set. We see that the vast majority of certificates is verified through a chain of length ≤ 3. For the EFF data, which was obtained in the first half of 2010, this cutoff is even at a length of ≤ 2. At the other end, more than half of the certificates have a chain length of 0. An explanation for this can be the relatively large fraction of self-signed certificates. When comparing Tue-Nov2009 to TUM-Apr2011, we see that the share of chains of length 0 has greatly decreased while the share of chains with length ≥ 1 has significantly increased by about 20%. The graph, as well as the increased average chain length (0.52 for Tue-Nov2009 vs. 0.67 for TUM-Apr2011) point to a weak tendency in the past 1.5 years to employ more intermediate certificates, not less. Overall, however, certification chains have remained remarkably short. Considering the trade-off of too many intermediate certificates vs. the benefits of using them, this seems more a positive development than a negative one.

The maximum length of chains found in scans is 16-17; in the EFF data, it is 18; whereas in the monitoring data, it is only 11 and 12, respectively. The scans thus have detected hosts with very unusual chain lengths (outliers), whereas most certificate chains are actually relatively short. In summary, we note that the chain length distributions of the four data sets do not exhibit very large differences.

Concerning how often unneeded self-signed certificates were included in the server's response, we only found 1,727 occurrences

[2]The minimum chain length was naturally 0, as some end-host certificates are self-signed or do not require intermediate certificates.

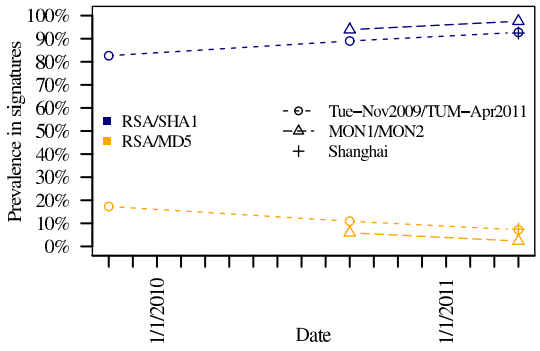

Figure 10: Popular signature algorithms in certificates.

in Tue-Nov2009 and 2,057 in TUM-Apr2011 where this was the case. As this is an unexpectedly low number, we also checked the EFF data set. Here, we find 63,290 occurrences in 4,375,901 valid chains. Our conclusion is that unneeded self-signed intermediate certificates in chains are rare. This is a good finding for performance.

Signature Algorithms. Cryptographic hash algorithms have come under increasing pressure in the past years, especially MD5 [28]. Even the stronger SHA1 algorithm is scheduled for phase-out [29]. When researchers announced in 2008 they could build a rogue CA using MD5 collisions [30], a move away from MD5 was expected to begin.

We thus extracted the signature algorithms in certificates. Figure 10 shows the results for TUM-Apr2011 and Tue-Nov2009, MON1 and MON2 (for all certificates and distinct ones), and Shanghai. We have omitted the rare cases (less than 20 occurrences altogether) where we found the algorithms GOST, MD2 and SHA-512 (a variant of SHA). The algorithms SHA256 and SHA1 with DSA were also only rarely found, in 0.1% of cases or less.

In our active scans, 17.3% of certificates were signed with a combination of MD5 and RSA in 2009. In 2011, this has decreased by 10% and SHA1 has risen by about the same percentage. The view from the monitoring data was slightly different. Most importantly, MD5 usage numbers were lower, both for all certificates and only for distinct ones. Between September 2010 and April 2011, the number had even fallen further.

Our conclusion here is that while MD5 is still sometimes used, it is indeed being phased out and does not play an import role in this PKI any longer.

Public Key Properties. It is quite evident that the ciphers used in certificates should be strong, and keys should be of a suitable length to achieve the desired security. Otherwise the certificate might be crackable by an attacker: RSA with 768 bit was factored in 2009 [31]. With the security margin of RSA-1024 shrinking, a move towards longer ciphers was recommended [32]. Furthermore, there should be no duplicate keys between different certificates, at least if the key owners are not the same.

We thus investigated the public keys in certificates. Concerning the ciphers, the result is very indicative. In the active scans Tue-Nov2009 and TUM-Apr2011, which span 1.5 years, the percentage of RSA keys on all queried hosts was always around 99.98%. DSA keys made up the rest. Counting only distinct certificates, the percentages remained the same. The values for the monitoring runs are practically identical. In these two scans, we also found a movement towards longer RSA key lengths: the percentage of

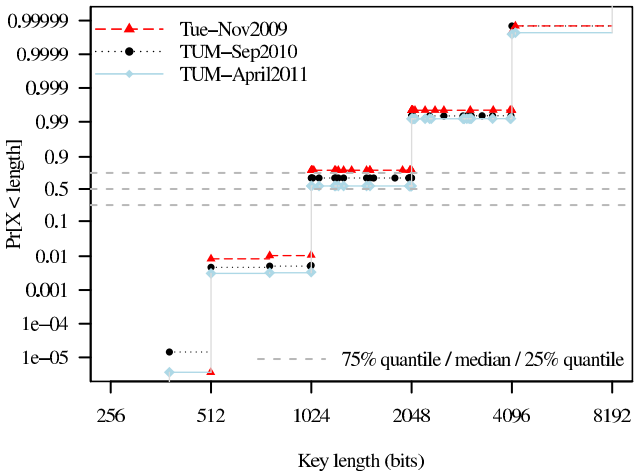

Figure 11: Cumulative distribution of RSA key lengths. (Note the unusual atanh scaling (double-ended pseudo-log) of the *y* axis.)

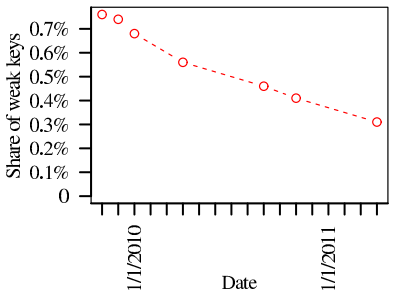

Figure 12: Debian weak keys over the course of 1.5 years in the scans of the Alexa top 1 million.

keys with more than 1,024 bit increased by more than 20% while the percentage of 1,024 bit keys fell by about the same. The general trend towards longer key lengths can be seen in Fig. 11: the newer the data set, the further the CDF graph is shifted along the *x* axis. This shows that the share of longer key lengths has increased while shorter key lengths have become less popular, with the notable exception of 384 bit keys that were found in the crawls from 2010 and 2011, but not in the 2009 data (left-hand side). The small triangles/circles/lozenges along the curves indicate the jumps of the CDF curves; hence they reveal furthermore that there is a significant number of various non-canonical key lengths, i.e., key lengths that are neither a power of 2 (e.g., 2,048) nor the sum of two powers of 2 (e.g., 768). However, their share is extremely small, as the CDF lines do not exhibit any significant changes at these locations.

An exponent and modulus make up the public key together; and there is only one private key for every public key. Concerning RSA exponents, the most frequent RSA exponent we found in Tue-Nov2009 was 65,537, which accounts for 99.13% of all exponents used. The next one was 17, which accounts for 0.77% of exponents.

There are two caveats to watch out for in public/private keys. The first refers to the Debian OpenSSL bug, which we have described in Section 3. We determined the number of certificates with weak keys of this kind by comparing with the official blacklists that come with every Debian-based Linux distribution. Figure 12 shows the results for our scans of the last 1.5 years. We can see that the number of affected certificates is clearly diminishing, and our numbers fit well with where the investigation in [14] left off. Furthermore,

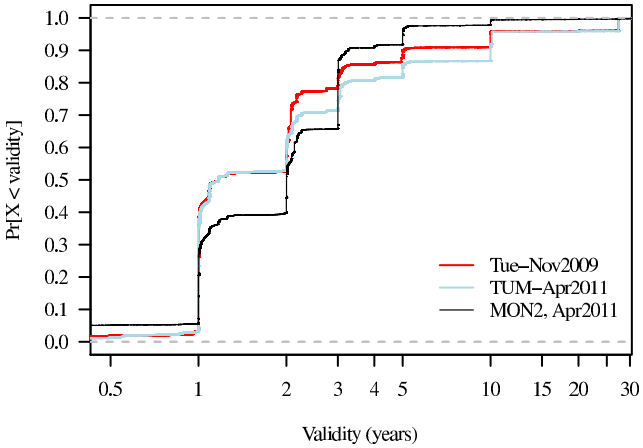

Figure 13: Comparison of certificate validity periods.

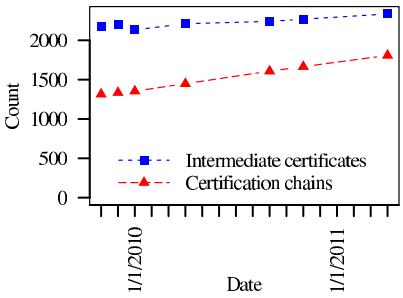

Figure 14: Temporal development of the number of distinct intermediate certificates (squares) and the number of distinct certification chains (triangles) across the scanning data sets.

the percentage of affected certificates was four times less in our monitoring data (about 0.1%) – this is a very good finding, as it shows that the servers that most users deal with are generally unlikely to be affected. However, we also found that a number of weak yet absolutely valid certificates (correctly issued by CAs, correct host names etc.) is still in use, but the number is very small: such certificates were found on about 20 hosts in the last scan.

The second caveat is that no combination of exponent and modulus should occur twice in different certificates. However, in TUM-Apr2011 we found 1,504 distinct certificates that shared the public/private key pair of another, different certificate. In Tue-Nov2009, we found 1,544 such certificates. The OpenSSL bug could be a cause for this, but we found that only 40 (Tue-Nov2009) and 10 (TUM-Apr2011) of the certificates fell into this category. We found one possible explanation when we looked up the DNS nameservers of the hosts in question. In about 70% of cases, these pointed to nameservers belonging to the same second-level domain. In about 28% of cases, these domains were `prostores.com` and `dnsmadeeasy.com`. We can only offer a hypothesis here, namely that some Web space providers issue a kind of default certificate and reuse the public/private keys. We do not know a reason for this practice, but it does not seem commendable: it means some private key owners are theoretically able to read encrypted traffic of others. They can even use a scanner to find out which domain to target.

Our conclusion here is that shorter key lengths are still encountered too often, but the trend is very positive. RSA keys should not be issued anymore at less than 2,048 bit. The Debian OpenSSL vulnerability has become rare in certificates. The cases of duplicate keys are curious, but not very frequent.

Validity Period. Certificates contain information for how long they are valid. This validity period is also a security factor. If a certificate is issued for a very long period, advances in cryptography (especially hash functions) can make it a target for attack.

When we analyzed the validity period for the certificates we encountered in our scans, we found that the majority of the certificates is issued with a life span of 12–15 months, i.e., one year plus a variable grace period. Other popular lifespans are two years, three years, five years, and ten years. This can be seen from the cumulative distribution function of the certificate life spans depicted in Fig. 13. Comparing Tue-Nov2009 to TUM-Apr2011, we can see that the share of certificates with an expiry time of more than 2 years has increased, in particular the share of certificates lasting 10 years. The curve for MON2 reveals that that the life spans typically encountered by users is either one, two, or three years, plus some grace period. In particular, certificates with life spans of more than five years seem to be used only rarely. What the figure does not show are the extremal values for the certificate life span: we encountered life spans on a range from two hours up to an optimistic 8,000 years. On the whole, however, the validity periods of certificates do not seem to be a major cause for concern.

Intermediate Certificates and Certification Chains. As explained in Section 2, many CAs do not issue certificates directly from their Root Certificates, but from an intermediate certificate (or a series thereof) that is signed by their Root Certificate. While beneficial for security, it also means that if too many intermediate certificates are used in a chain, the undesired result can be that the attack surface increases, as there are more keys that can be attacked. This is particularly true if intermediate certificates are accessible by regional resellers of CAs. We thus investigated the number of distinct intermediate certificates and the distinct certification chains that are built with them.

Figure 14 shows the development of the intermediate certificates. For the active scans from Germany, we see about 2,300, with a trend to increase. Compared to the size of the Root Store in Firefox, this means that, on average, every CA would use more than 10 intermediate certificates. However, we already know that the average chain lengths are much smaller, so this points to a very skewed distribution. The number of distinct intermediate certificates in the EFF data set is even higher, almost grotesque: 124,387. The ratio of certificates/intermediate certificates for the top 1 million hosts is about 335 (TUM-Apr2011); in contrast, it is about 91 for the whole IPv4 space. This means that the top 1 million hosts use less intermediate certificates than hosts in the whole IPv4 space do.

To analyze chains, we computed a unique ID for every distinct certification chain we found. For this, we discounted any self-signed intermediate certificate in a chain as a potential Root CA. The remaining intermediate certificates were sorted, concatenated and hashed. Recall that the number of intermediate certificates is very small compared to the number of end-host certificates. Correspondingly, we found only a small number of certification chains. This means that the X.509 certification 'tree' (end-hosts are leaves) shrinks rapidly from a wide base to very few paths leading to the Root CAs. In the EFF data set, we find an unexpected number of different chains: 17,418, which is much higher than the number in our scans. The number of distinct intermediate certificates in the active scans was always close in magnitude to the number of distinct certification chains (Fig. 14), whereas for the EFF data, the

Scan	*Suspicious certificates*	*Differences to TUM-Apr2011*
Santa Barbara	1,628	5,477
São Paulo	1,643	6,851
Melbourne	1,824	7,087
İzmir	2,069	7,083
Boston	2,405	5,867
TUM-Apr2011	3,245	—
Shanghai	10,194	9,670
Bejing	10,305	9,901
Moscow	10,986	11,800

Table 4: Occurrences of suspicious certificates per location, and number of certificates different to those seen from TUM.

number of different chains is smaller by a factor of 10. This indicates that the convergence to very few certification paths is much more expressed for the global IPv4 space than for the popular Alexa hosts. In other words, there is more relative variance in Root CAs in the top 1 million than in the whole IPv4 space.

Overall, our finding here is that too many intermediate certificates are encountered (leading to an unnecessary attack surface), even though it seems they are usually not needed for the (shorter) certification chains. In the top 1 million hosts, the situation is better than in the IPv4 space as a whole.

Different Certificates between Locations. We investigated how many downloaded certificates were different between locations. Our motivation was two-fold. A Web site operator may offer duplicated or different content depending on geographic region; Content Distribution Networks (CDNs) are a large-scale example of this. In this case, it is possible that certificates are bought from different CAs. It is also possible that an operator has switched CAs but not propagated the move to all sites yet.

Another possible reason can be of a malicious nature: a router can intercept traffic and swap certificates transparently on the fly. Through this Man-in-the-middle attack, the attacker knows the private key of the swapped certificate and is thus able to read the encrypted traffic. We labeled hosts as 'suspicious' when their certificate was identical from most locations and only differed at 1–3 locations. Table 4 shows the results from each vantage point.

Although the suspicious cases seem to occur particularly often with respect to hosts scanning from China and Russia, this might as well be the result of localized CDN traffic. We thus examined differences between Shanghai and TUM-Apr2011. The number of different certificates between these two locations was 9,670, which is about 1% of all certificates in TUM-Apr2011. Only 213 of the corresponding sites were in the top 10,000; the highest rank was 116. We can surmise that if operators of high-ranking sites use CDNs for their hosting in these regions, then they correctly deploy the same certificates. From manual sampling, we could not find a Man-in-the-middle interception from our vantage points.

We checked how many of the certificates from Shanghai were actually valid (correct chain, CN etc.). This yielded just 521 cases – and only in 59 cases, the certificate was absolutely valid in TUM-Apr2011 but not in Shanghai. We checked the corresponding domains manually; not one of them could be identified as highly sensitive (e.g., politically relevant, popular Web mailers, anonymization services). About a quarter of certificates were self-signed but different in Shanghai and TUM-Apr2011. The reason for this is unknown to us, but we do not think it is related to an attack.

While we are reluctant to offer compelling conclusions here, we do wish to state the following. First, there are not many differences between locations. High-ranked domains using CDNs seem to properly distribute their certificates. Maybe this is the most interesting finding, given the overall administrative laxness that is revealed by the many badly maintained certificates. Second, the number of different certificates was significantly higher from the vantage points in China and Russia. However, we found no indication of an attack. Rather, the findings seem to point at administrative difficulties. Third, we emphasize that we did not manually investigate all domains – we will thus publish the data sets where certificates were different so others can investigate them, too.

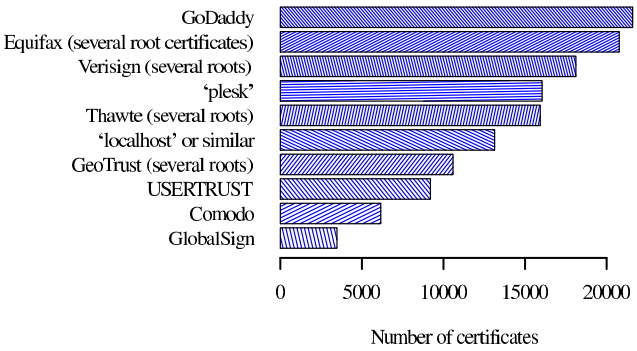

Figure 15: Top ten of issuers in TUM-Apr2011.

Certificate Issuers. We were interested to see which issuers occur most commonly, since the corresponding CAs are attack targets of higher value, due to the higher value of the customer database. We therefore determined the most common issuers for certificates in the data set TUM-Apr2011 (distinct certificates). There were 30,487 issuers in total.

Then, we concentrated on issuers that were found in at least 2,000 certificates. In total, these issuers accounted for 135,011 certificates – this is more than half of all distinct certificates. Figure 15 presents the top ten. Note, however, that there are close to 200 Root Certificates in the Firefox Root Store, and several CAs have actually more than one Root Certificate in there. Where possible, we thus identified issuers as companies with subsidiaries not grouped together. Obviously, two entries are not companies: as was explained before, 'plesk' as the issuer is an indication of virtualization, and `localhost` or similar entries were always issuers in self-signed certificates.

Further Parameters. We inspected the serial numbers in valid certificates and looked for duplicates by the same issuing CA. We did not find any in the last three scans, TUM-Apr2011, TUM-Nov2010 and TUM-Sep2010. This is a good finding, as a certificate's serial number is intended to be a unique identifier (in fact, blacklisting of certificates in so-called revocation lists is done via the serial numbers).

We also investigated a certificate property that is not security-relevant, but interesting nonetheless: X.509 Version 1 is outdated, and there is no reason to use it anymore. The current version is X.509 Version 3. We investigated data sets Tue-Nov2009 and TUM-Apr2011 in this regard. In November 2009, 86.01% of certificates were Version 3, 13.99% were Version 1. In April 2011, 85.72% were Version 3 and 14.27% Version 1.

Although our first guess was that this is an artifact of measurement, we found that 33,000 certificates with Version 1 had not been seen in any previous scan. None of them had valid certification chains, however, and 31,000 of them were self-signed. Of the others, the biggest issuer was a Russia-based company. We investi-

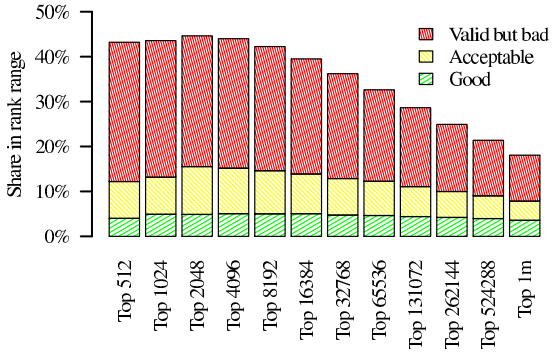

Figure 16: Certificate quality in relation to Alexa rank.

gated all issuers that had issued more than 2 certificates and found that all of them seemed to employ a PKI of their own, but without Root Certificates in Firefox. The reasons for this use of Version 1 are unknown to us, but plausible causes are software default settings or an arbitrary choice to use the simpler format of Version 1.

Certificate Quality: A Summarizing View.

We conclude our analysis with a summarizing view of certificate quality. Figure 16 reveals that the share of valid certificates (=total height of the bars) is negatively correlated with the Alexa rank. This does not come as a surprise, since operators of high-profile sites with a higher Alexa rank can be expected to invest more resources into a working HTTPs infrastructure. What is surprising, however, is that even in the top 512 or 1,024 – i.e., truly high-ranking sites – only just above 40% of certificates are absolutely valid. The figure also shows a classification of the certificates in three categories, which we have termed 'good', 'acceptable' and 'valid but bad'. *Good* certificates have correct chains, correct host names, a chain length of 2 at most, do not use MD5 as a signature algorithm, use non-weak keys that are either DSA keys or RSA keys of more than 1024 bit, and have a validity of at most 13 months (a year plus a grace period). For *acceptable* keys, we require the same but allow a chain length of 3 and validity is allowed to be up to 25 months. *Valid but bad* keys represent the remainder of keys (with correct chains and host names, but that otherwise fail our criteria). Interestingly, although high-profile sites are more likely to deliver valid certificates, the share of bad certificates among their valid certificates is much higher (about two thirds) when compared to that of all sites (about half). The conclusion we have to draw here is that even among the absolutely valid certificates the number of certificates with strong properties is disappointingly low.

6. DISCUSSION AND CONCLUSION

By combining and evaluating several actively and passively obtained data sets, which were in part also obtained over 1.5 years, we were able to derive a very comprehensive picture of the X.509 infrastructure as used with TLS/SSL. Our analysis supports with hard facts what has long been believed: that the X.509 certification infrastructure is, in great part, in a sorry state.

The most sobering result is probably the percentage of certificates that a client using the Mozilla Root Store would accept without a warning: just 18%. This can be traced to both incorrect certification chains (40% of all certificates exhibit this problem), but even more so to incorrect or missing host names in the subject or subject alternative name. With self-signed certificates, where conscientious operators would have an incentive to use the correct host name, the situation was much worse. The only positive point is that the percentages of absolutely valid certificates have increased since 2009, but then again only very slightly. Recall that these numbers refer to the top 1 million hosts – the percentage of certificates where the chains are correct is lower for the full IPv4 space than for the top sites, as we found by examining the EFF data set.

Moreover, many certification chains showed more than one error. Expired certificates were common, and so were certificates for which no Root Certificate could be found. A further problematic finding is that all our data sets reveal a high number of certificates that are shared between a large number of hosts. This is even the case for high-profile Web hosters – and often, the host names do not match the certificates there, either. Although offered by several CAs, Extended Validation certificates do no seem to be in wide use.

This truly seems a sorry state, and it does not come as a surprise that users just click away warnings, thus adding to the general insecurity of the WWW. As few CAs are responsible for more than half of the distinct certificates, one should think the situation should be better or at least easier to clean up.

There are some more positive tendencies that should be mentioned, however. Our evaluation shows that the more popular a given site is, the more likely it supports TLS/SSL at all, and the more likely it shows an absolutely valid certificate. On the whole, key lengths seem not to constitute a major problem. The same is true for signature algorithms. Keys with short bit lengths are becoming fewer, and the weak MD5 algorithm is clearly being phased out, too. Over the past 1.5 years, we also found an increase in the use of intermediate certificates while chain lengths remained remarkably short. This is a good development, as end-host certificates should not be issued by a Root Certificate that is used in online operations. However, use of intermediate certificates can be overdone and sometimes is. The latter will add to the already complex certification infrastructure that we have encountered, with a high number of distinct certification chains.

Concerning our passive monitoring, the data we obtained allowed us to evaluate negotiated properties of TLS/SSL associations, which cannot be obtained by active scans. We were able to determine the negotiated ciphers and digest mechanisms. Our observations lead us to conclude that most connections use secure ciphers with acceptable key lengths, and that key lengths are even increasing over time, with a good security margin. This is in line with the corresponding trends in certificate keys. However, MD5 is still commonly used to compute Message Authentication Codes (MACs). This is not really dramatic at the moment – even now, collision attacks take at least some seconds, and preimage attacks much longer. An attacker's time window is likely too short for a useful on-the-fly attack at this time (note that TLS/SSL uses a replay protection for the payload, too). However, we cannot see a compelling reason to continue using MD5 for MACs (except possibly for some very old legacy clients). We feel it should be discouraged and phased out.

With the above mentioned problems in certificates, however, we have to conclude that the positive movements do not address the most pressing problem, which is the certification structure itself. Unfortunately, the general state seems to persist as made evident by the fact that several critical factors like validity of certification chains, correct host names in certificates, or plainly the number of hosts that offer TLS/SSL do not seem to change. As this work has focused on the top 1 million hosts (scans) and the PKI as accessed by users (monitoring), this seems to indicate a pressing need.

7. ACKNOWLEDGEMENTS

We thank Nathan Evans and the Leibniz-Rechenzentrum (LRZ) in Munich for assisting us with obtaining the data. We especially

thank Helmut Reiser of LRZ for his valuable feedback. This work has been partially funded by the European Union within the EU FP7 project ResumeNet (FP7-224619).

8. REFERENCES

[1] D. Cooper, S. Santesson, S. Farrell, S. Boeyen, R. Housley, and W. Polk, "Internet X.509 public key infrastructure certificate and certificate revocation list (CRL) profile," RFC 5280 (Proposed Standard), May 2008.

[2] T. Dierks and E. Rescorla, "The transport layer security (TLS) protocol version 1.2," RFC 5246 (Proposed Standard), Aug. 2008, updated by RFCs 5746, 5878, 6176.

[3] J. Appelbaum, "Detecting certificate authority compromises and Web browser collusion," Blog entry: https://blog.torproject.org/blog/detecting-certificate-authority-compromises-and-web-browser-collusion, 2011, [online; last retrieved in May 2011].

[4] Mozilla Security Blog, "DigiNotar removal follow up," https://blog.mozilla.com/security/2011/09/02/diginotar-removal-follow-up/ [online; last retrieved in September 2011], 2011.

[5] C. Herley, "So long, and no thanks for the externalities: the rational rejection of security advice by users," in *Proc. 2009 Workshop on New Security Paradigms*. New York, NY, USA: ACM, 2009.

[6] C. Ellison and B. Schneier, "Ten risks of PKI: What you're not being told about public key infrastructure," *Computer Security Journal*, vol. 16, no. 1, 2000.

[7] P. Gutmann, "PKI: It's not dead, just resting," *IEEE Computer*, vol. 35, no. 8, August 2002.

[8] P. Eckersley and J. Burns, "An observatory for the SSLiverse," Talk at Defcon 18., July 2010, [last retrieved in May 2011]. [Online]. Available: https://www.eff.org/files/DefconSSLiverse.pdf

[9] P. Eckersley and J. Burns, "Is the SSLiverse a safe place?" Talk at 27C3. Slides from https://www.eff.org/files/ccc2010.pdf [online; last retrieved in May 2011], 2010.

[10] I. Ristic, "Internet SSL Survey 2010," Talk at BlackHat 2010. Slides from https://media.blackhat.com/bh-us-10/presentations/Ristic/BlackHat-USA-2010-Ristic-Qualys-SSL-Survey-HTTP-Rating-Guide-slides.pdf, 2010, [online; last retrieved in May 2011].

[11] I. Ristic, "State of SSL," Talk at InfoSec World 2011. Slides from http://blog.ivanristic.com/Qualys_SSL_Labs-State_of_SSL_InfoSec_World_April_2011.pdf, 2011, [online; last retrieved in May 2011].

[12] Alexa Internet Inc., "Top 1,000,000 sites (updated daily)," http://s3.amazonaws.com/alexa-static/top-1m.csv.zip, 2009–2011, [online; last retrieved in May 2011].

[13] H. K. Lee, T. Malkin, and E. Nahum, "Cryptographic strength of SSL/TLS servers: Current and recent practices," in *Proc. 7th ACM SIGCOMM Conference on Internet Measurement (IMC)*, San Diego, CA, USA, October 2007.

[14] S. Yilek, E. Rescorla, H. Shacham, B. Enright, and S. Savage, "When private keys are public – results from the 2008 Debian OpenSSL vulnerability," in *Proc. 9th ACM SIGCOMM Conference on Internet Measurement (IMC)*, Chicago, Illinois, USA, Nov. 2009.

[15] Data sets of active scans, http://pki.net.in.tum.de, 2011.

[16] A. Croll and S. Power, *Complete Web Monitoring*. O'Reilly Media, 2009.

[17] L. Braun, G. Münz, and G. Carle, "Packet sampling for worm and botnet detection in TCP connections," in *Proc. IEEE/IFIP Network Operations and Management Symposium (NOMS)*, Apr. 2010.

[18] S. Kornexl, V. Paxson, H. Dreger, A. Feldmann, and R. Sommer, "Building a time machine for efficient recording and retrieval of high-volume network traffic," in *Proc. 5th ACM SIGCOMM Conference on Internet Measurement (IMC)*, Berkeley, CA, USA, Oct. 2005.

[19] L. Braun, A. Didebulidze, N. Kammenhuber, and G. Carle, "Comparing and improving current packet capturing solutions based on commodity hardware," in *Proc. 10th ACM SIGCOMM Conference on Internet Measurement (IMC)*, Nov 2010.

[20] F. Fusco and L. Deri, "High speed network traffic analysis with commodity multi-core systems," in *Proc. 10th ACM SIGCOMM Conference on Internet Measurement (IMC)*, Nov 2010.

[21] V. Paxson, "Bro: a system for detecting network intruders in real-time," *Computer networks*, vol. 31, no. 23-24, 1999.

[22] H. Dreger, A. Feldmann, M. Mai, V. Paxson, and R. Sommer, "Dynamic Application-Layer Protocol Analysis for Network Intrusion Detection," in *Proc. USENIX Security Symposium*, Apr. 2006.

[23] Planet Lab, "Planet Lab Web site," https://www.planet-lab.org [online; last retrieved in May 2011].

[24] The International Grid Trust Federation, "IGTF Web site," http://www.igtf.net/ [online; last retrieved in May 2011].

[25] A. Klein, "Attacks on the RC4 stream cipher," *Designs, Codes and Cryptography*, vol. 48, 2008.

[26] E. Rescorla, "HTTP over TLS," RFC 2818 (Informational), 2000.

[27] CA/Browser Forum, "EV SSL certificate guidelines version 1.3," http://www.cabforum.org/Guidelines_v1_3.pdf, 2010, [online; last retrieved in May 2011].

[28] M. Stevens, A. Lenstra, and B. de Weger, "Chosen-prefix collisions for MD5 and colliding X.509 certificates for different identities," in *Advances in Cryptology – EUROCRYPT 2007*, ser. LNCS. Springer Berlin / Heidelberg, 2007, vol. 4515.

[29] NIST, "Approved Algorithms," http://csrc.nist.gov/groups/ST/toolkit/secure_hashing.html, 2006, [online; last retrieved in May 2011].

[30] A. Sotirov, M. Stevens, J. Appelbaum, A. Lenstra, D. Molnar, D. A. Osvik, and B. de Weger, "MD5 considered harmful today," http://dl.packetstormsecurity.net/papers/attack/md5-considered-harmful.pdf, 2008, [online; last retrieved in May 2011].

[31] T. Kleinjung, K. Aoki, J. Franke, A. Lenstra, E. Thomé, J. Bos, P. Gaudry, A. Kruppa, P. Montgomery, D. Osvik, H. te Riele, A. Timofeev, and P. Zimmermann, "Factorization of a 768-bit RSA modulus," in *Advances in Cryptology – CRYPTO 2010*, ser. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, 2010, vol. 6223.

[32] NIST, "Special publications (800 Series)," http://csrc.nist.gov/publications/PubsSPs.html, 2011, [online; last retrieved in May 2011].