TLS Proxies: Friend or Foe?

Mark O'Neill, Scott Ruoti, Kent Seamons, Daniel Zappala
Brigham Young University, Department of Computer Science
mto@byu.edu, ruoti@isrl.byu.edu, seamons@cs.byu.edu, zappala@cs.byu.edu

ABSTRACT

We measure the prevalence and uses of TLS proxies using a Flash tool deployed with a Google AdWords campaign. We generate 2.9 million certificate tests and find that 1 in 250 TLS connections are TLS-proxied. The majority of these proxies appear to be benevolent, however we identify over 1,000 cases where three malware products are using this technology nefariously. We also find numerous instances of negligent, duplicitous, and suspicious behavior, some of which degrade security for users without their knowledge. Distinguishing these types of practices is challenging in practice, indicating a need for transparency and user awareness.

1. INTRODUCTION

Secure communication on the Internet is based primarily on digital certificates signed by certificate authorities and intermediate authorities. This validation system is currently being compromised by the use of TLS proxies, which can act as a man-in-the-middle (MitM) for TLS connections [3]). A TLS proxy can issue a *substitute certificate* for any site the user visits, so that the user establishes an encrypted connection to the proxy rather than the desired web site. The proxy can then decrypt and monitor or modify all user traffic, before passing it along via a second encrypted channel to the desired web site. TLS proxies are used for a variety of legitimate purposes, such as blocking malware, but can also be used by malicious entities to compromise the privacy or security of end users. Isolated attacks have been observed in the wild, notably in Iran [1] and Syria [7]. The most dangerous aspect of TLS proxies is that the user is entirely unaware that encrypted traffic is being intercepted by an organization or attacker; browser software still shows a lock icon during such sessions. Thus TLS proxies are controversial because they mislead users and compromise the end-to-end security promises made by TLS.

Detecting the presence and prevalence of TLS proxies is a challenging measurement problem. To detect a proxy, we must obtain the certificate a client, such as a web browser, actually obtains, then compare this with the valid certificate presented by the server the client is contacting. A mismatch indicates that some kind of proxy, either benevolent or malicious, is intercepting the client's traffic to that particular server. To determine the prevalence of TLS proxies, we must repeat this measurement on as many client systems as possible.

Two recent works have found some evidence for TLS proxies by measuring certificates received by clients. Huang et al. measure the prevalence of TLS proxies that intercept traffic from clients connecting to Facebook [9], finding that 1 in 500 TLS connections are proxied, mostly by corporate Internet filters and personal antivirus software. In addition, 1,112 connections were found to be intercepted by malware. Because this study uses Flash to detect a certificate mismatch, it does not detect proxies affecting most mobile devices. The Netalyzer project measured certificates received by Android apps, assessing 15,000 sessions and identifying just one case of a TLS proxy running in an analytics app [22]. Though this is a very low rate of prevalence (30 times less than Huang's study), the app was found to whitelist several sites, including Facebook. This indicates that measurements of proxies should examine low-profile sites that are unlikely to be whitelisted.

To measure the prevalence of proxies we use a Flash app deployed with a Google AdWords campaign. Like Huang, our measurements use Flash to detect a certificate mismatch without any user interaction. However, the deployment via Adwords affords some advantages. First, we can actively measure clients, based on how much money we spend on the advertisement, enabling us to collect 12 million of measurements in a week by spending $750 per day. Second, we target our measurements toward a server that ordinarily does not receive significant traffic. This enables detection of proxies that intentionally whitelist popular sites such as Facebook.

IMC 2016, November 14 - 16, 2016, Santa Monica, CA, USA

 ISBN 978-1-4503-4526-2/16/11…$15.00

DOI: http://dx.doi.org/10.1145/2987443.2987488

Using the AdWords campaign, we were able to test 2.9 million connections for substitute certificates. Our findings are as follows:

- We found 11,764 proxied connections out of 2.9 million total measurements (0.41% or approximately 1/250 of all connections) spanning 142 countries. This rate is double that reported by Huang, which provides evidence that some proxies may use whitelisting. We found that most substitute certificates claim to be from benevolent TLS proxies, with 70.87% claiming to be generated by a firewall software and 12.66% claiming to be generated by a corporate network.
- We found over 2,000 instances of negligent and malicious behavior. Our analysis of one parental filter found that it masks forged certificates, allowing an attacker to easily perform a MitM attack against the firewall's users. In addition, we found three malware products affecting over 1,000 connections that install a new root certificate and act as a TLS proxy to dynamically insert advertisements on secure sites. We also found evidence that spammers are using TLS proxies in their products. We found numerous other suspicious circumstances in substitute certificates, such as a null Issuer Organization, falsified certificate authority signatures, and downgraded public key sizes.

2. BACKGROUND

To validate the identity of a website such as Amazon, the web browser relies on certificate authorities (CAs), which digitally sign certificates vouching for the identity of the web server. When the browser initiates a TLS connection with a server, it retrieves the server's certificate, then must verify the certificate's validity before exchanging encrypted traffic.

Web browsers authenticate a site by validating a chain of digital signatures from the site's certificate back to one of a set of trusted root certificates. These certificates comprise the "root store" and are typically bundled with the operating system or browser. For example, the certificate for `www.google.com` is signed by the Google Internet Authority G2, an intermediate certificate authority run by Google. This certificate is in turn signed by GeoTrust Global CA, a certificate authority whose certificate is located in the root store of the browser or operating system. A substitute certificate's signature should not be able to be traced back to a root store certificate and should be rejected.

This system can be attacked by a TLS proxy inserting itself as a man-in-the-middle between the browser and the web server. As shown in Figure 1, when the browser tries to open a secure connection to the web server, this connection is instead intercepted by the proxy. The proxy also provides a falsified, substitute certificate to the browser, so that it can impersonate the original

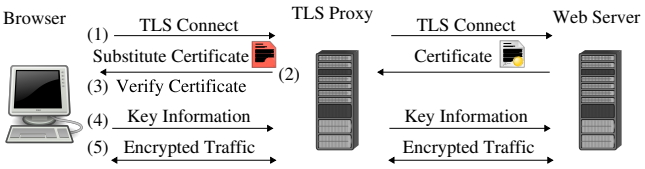

Figure 1: "Secure" session establishment with involving a TLS proxy

website. For this to work, the proxy must somehow control a substitute certificate for the original website that validates against the root store of the user. This can be accomplished in a variety of ways, both benign and malicious.

Benign ways to provide valid substitute certificate include (a) using enterprise software to supply certificates for the root store of all computers in an organization, (b) creating a software image using new root certificates, or (c) installing a new root certificate with software such as a personal firewall. Generally these methods are used for benevolent reasons, such as blocking malware and viruses, providing intrusion detection, or protecting intellectual property.

Other ways of providing seemingly-valid certificates are more nefarious. For example, malware typically has permission to add new root certificates when it is installed inadvertently by the user. Alternatively, a rogue certificate authority can issue any certificate it wants, since all root certificates are allowed to sign for any domain. There have also been numerous reported cases of compromised and negligent certificate authorities that allow attackers to issue fraudulent certificates [6]. In addition, governments have the ability to coerce authorities into granting them substitute certificates.

3. MEASUREMENT TOOL

We have developed a tool to measure the prevalence of TLS proxies using existing, widely-deployed technologies. The tool runs silently from the perspective of the user, with no user action required to install or run it.

3.1 Design

Our tool works as shown in Figure 2. The client browser connects to an ad server where the Flash application is hosted. The application is embedded in an advertisement, which is downloaded and automatically run by the browser. The tool sends a `ClientHello` message to the Web Server to initiate a TLS handshake. The tool then records the `ServerHello` and `Certificate` messages received in response and terminates the handshake. The retrieved certificates are then forwarded to the Database Server for analysis.

To measure a TLS proxy, the Web Server must host a simple *socket policy file*. For security reasons Flash 9.0 and above requires that applications attempting to establish a TCP connection with a remote host first obtain permission from that host via this policy file.

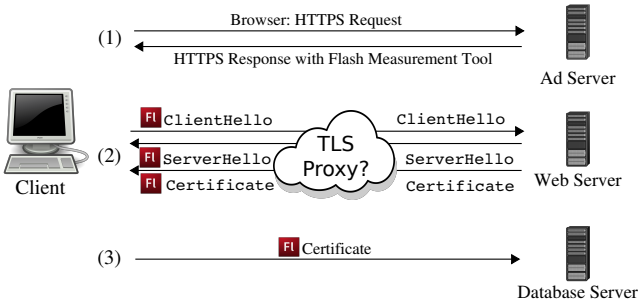

Figure 2: Using an Ad Server for TLS Proxy Measurement

Our Web Server's socket policy file is served on port 80. This avoids effects of captive portals, which often block traffic targeting ports other than those used by HTTP and HTTPS (e.g., airport public access WiFi).

3.2 Implementation

To implement our tool it was necessary to retrieve the certificate used during a TLS handshake. It would have been preferable to use JavaScript or HTML5 to retrieve the certificate used as part of a current TLS connection, but unfortunately there is no API available for this. Firefox allows a plugin to request the certificate, but plugins require manual client installation. This left us with the alternative of establishing a plain TCP connection with the target server and then initiating a TLS handshake. Unfortunately, the ability to use a plain TCP connection rules out the use of HTML5 `WebSockets`.

Due to these constraints, we opted to use the Adobe Flash platform. We implemented our tool in ActionScript using only libraries supported by the Flash 9.0 runtime, due to its nearly complete market penetration relative to newer versions. Using the `Socket` API provided by Flash 9.0 we implemented functionality required to perform a partial TLS handshake. After receiving the full `Certificate` message from the Web Server the handshake is aborted and the connection is closed. The Flash application records and parses all certificates received from the `Certificate` message (as some hosts offer certificate chains) and stores them locally until it parses the final one. All certificate data, in PEM format, is concatenated and then sent as an HTTP POST request to the Database Server for analysis.

Code and collected datasets are available for download at https://tlsresearch.byu.edu.

3.3 Limitations

Our tool is unable to measure TLS proxies being used against most mobile devices. An overwhelming majority of mobile platforms do not support Flash, and Adobe has discontinued their development of Flash for mobile devices.

It is possible that TLS proxies could be engineered to circumvent our measurements. At the time of our study, our measurement methodology was not well known, so it is unlikely that any attacker was evading detection or tampering with our reports. However, in the case that this methodology becomes well-known, it would be difficult to prevent dedicated attackers from modifying their TLS proxies to avoid our measurements.

While our tool is capable of using multiple hosts simultaneously as the "Web Server", each of these hosts must serve a socket policy file (described previously) that allows this remote connection to occur.

4. GOOGLE ADWORDS CAMPAIGN

To achieve rapid and widespread deployment of our measurement tool we leveraged the Google AdWords platform. This strategy for using an advertising campaign to conduct an end-user measurement study has previously been used to study CSRF attacks [2], DNS rebinding attacks [11], and DNSSEC deployment [10, 13]. Our study is the first to use this same method to measure the deployment of TLS proxies.

4.1 Deployment

The deployment of the measurement tool is given in Figure 2. Deployment responsibilities were delegated to the Google AdWords platform, while all reports from the tool were sent back to a Database Server we controlled. To accommodate placement in advertisements, our measurement tool was modified to contain a visible canvas on which we place a simplistic advertisement for our research lab. Our measurement tool was run as soon as the user's browser loaded the advertisement, and required no interaction from the users.

For our ad campaign we leveraged the CPM (cost-per-impression) bidding model for our campaign, which maximizes the number of unique clients presented with our ad. We set the Max. CPM to $10 USD. To help us reach a global audience we indicated that our ad should be served to all locations and languages. Additionally, since ads are shown only on websites that match a set of designated keywords we selected our keywords based on phrases that were currently trending globally on Google Trends[1]. We set our ad to show uniformly throughout the day so as to collect data from users in a variety of locations and situations (e.g., home, commuting, work).

Along with the certificate, we also recorded the IP address of the client tested. This IP address was then used to query the MaxMind GeoLite [17] database to gather geolocation information.

Our Google AdWords advertising campaign ran from January 6, 2014 to January 30, 2014. During the first 17 days of the study we varied the amount of money allocated to the ad campaign, but for the last week we kept it at $500/day. In this study we only gathered certificate data for our own website, *tlsresearch.byu.edu*. We used the following keywords for the study: *Nelson Mandela*, *Sports*, *Basketball*, *NSA*, *Internet*, *Freedom*,

[1]http://www.google.com/trends/?geo

Paul Walker, *Security*, *LeBron James*, *Haiyan*, *Snowden*, *PlayStation 4*, *Miley Cyrus*, *Xbox One*, *iPhone 5s*.

This campaign generated 4,634,386 impressions and 3,897 clicks (not required to complete the measurement) at a cost of $4,911.97. In total we completed 2,861,244 successful measurements.

4.2 Ethical Considerations

Our experimental design limits potential harm to users. Our tool makes several request to a web server we control, which is largely indistinguishable from the traffic generated by advertisements that load images, audio, or video from a web server. If network administrators were to investigate this traffic by visiting our web server, they would be shown a description of our research and provided with our contact information[2]. Our tool also logs the substitute certificate a browser sees, which some organizations or individuals may consider a privacy leak; this is balanced by only examining modifications made to a connection to our own web server and the potential benefits that come from viewing certificate information to identify malware. This and Huang's work [9] has informed the community about these practices and motivated a subsequent study that identified weaknesses in personal firewalls [5]. Other than the certificate, we only record data that is present in the web server's connection log. Finally, our tool complies with Google AdWords' terms of service.

5. RESULTS

During the ad campaign, we served 4.36 million ads and successfully completed 2.86 million measurements. Of those tests, 11,764 returned a different X.509 certificate than was served by our secure web server, indicating the presence of a TLS proxy.

The users behind a proxied connection that were identified by our campaign originated in 142 countries and from 8,589 distinct IP addresses. Due to the targeting algorithms used by Google AdWords, our tool's exposure to these countries is not uniformly distributed. Table 1 shows the countries with the most proxied connections in our study. For each country, the table lists the total number of proxied connections, the total number of connections, and the percentage of total connections to that country that were proxied. Some countries have significantly higher percentages of proxied connections than the average, including France (1.09%), Canada (0.87%), Belgium (0.81%), the United States (0.79%), and Romania (0.74%). Together, connections from the United States and Brazil account for 36% of detected proxies.

Rank	Country	Proxied	Total	Percent
1	France	812	74,789	1.09%
2	Canada	303	34,695	0.87%
3	Belgium	136	16,816	0.81%
4	US	2,252	285,078	0.79%
5	Romania	696	94,116	0.74%
6	Brazil	2,041	298,618	0.68%
7	Portugal	185	29,799	0.62%
8	India	302	51,348	0.59%
8	Turkey	303	65,195	0.46%
13	S.Korea	196	46,660	0.42%
14	Russia	224	58,402	0.38%
15	Spain	226	62,569	0.36%
16	Japan	111	31,751	0.35%
17	Netherlands	104	31,938	0.33%
18	UK	759	259,971	0.29%
19	Germany	499	187,805	0.27%
20	Ukraine	160	61,431	0.26%
21	Taiwan	101	61,195	0.17%
22	Poland	182	110,550	0.16%
23	Italy	200	129,358	0.15%
	Other (215)	1,972	869,096	0.23%
	Total	11,764	2,861,180	0.41%

Table 1: Proxied connections by country, ordered by percentage proxied

5.1 Analysis of Issuers

We first analyze the contents of the Issuer Organization in the substitute certificates we collected. We use OpenSSL to decode the certificates and store them in a database, where we can run queries. We also manually inspect the contents of the relevant fields to identify the issuing organization and their software products, using web searches to determine their identity. We emphasize that our results in this section are based on the intercepting proxy self-identifying themselves in the certificate. It is certainly possible that malicious proxies have hidden their tracks by masquerading as a legitimate organization in the Issuer Organization field, and we cannot detect this.

Table 2 shows the values for the Issuer Organization field of the substitute certificates. Table 3 provides a breakdown of values present in the Issuer Organization field of the substitute certificates. The majority of certificates from proxied connections have an Issuer Organization field matching the name of a personal or enterprise firewall (69.54%). Another 12.66% have the name of an organization set as the Issuer Organization (e.g., Lawrence Livermore National Laboratory, Lincoln Financial Group). Additionally, 7% (829) of the substitute certificates have null values for the Issuer Organization field.

The most suspicious activities discovered were revealed by certificates with an Issuer Organization that matched the names of malware. "Sendori, Inc," "WebMakerPlus Ltd," and "IopFailZeroAccessCreate" appeared in 966, 95, and 21 Issuer Organization fields, respectively. Sendori poses as a legitimate enterprise, however they produce software that compromises the DNS lookup of infected machines, allowing them to redirect users to improper

[2]We were never contacted.

Rank	Issuer Organization	Connections
1	Bitdefender	4,788
2	PSafe Tecnologia S.A.	1,200
3	Sendori Inc	966
4	ESET spol. s r. o.	927
5	Null	829
6	Kaspersky Lab ZAO	589
7	Fortinet	310
8	Kurupira.NET	267
9	POSCO	167
10	Qustodio	109
11	WebMakerPlus Ltd	95
12	Southern Company Services	62
13	NordNet	61
14	Target Corporation	52
15	DigiCert Inc	49
16	ContentWatch, Inc.	42
17	NetSpark, Inc.	42
18	Sweesh LTD	39
19	IBRD	26
20	Cloud Services	23
	Other (332)	1,121

Table 2: Issuer Organization field values

Proxy Type	Connections	Percent
Business/Personal Firewall	8,101	68.86%
Organization	1,394	12.66%
Malware	1,112	8.65%
Unknown	840	7.14%
Parental Control	156	1.33%
Business Firewall	69	0.59%
Certificate Authority	49	0.42%
School	32	0.27%
Personal Firewall	11	0.09%
Telecom	0	0%

Table 3: Classification of claimed issuer, ordered by percentage proxied

hosts. A TLS proxy component is used to bypass host authenticity warnings in the browser. The substitute certificates generated by the TLS proxy are signed by a root authority that was added to the root store of the local machine at the time of infection. Substitute certificates issued by Sendori originated from 30 distinct countries.

The WebMakerPlus malware is primarily associated with inserting advertisements into Web pages. We hypothesize that WebMakerPlus uses a TLS proxy to simulate that their advertisements are served from a secure connection and to modify secure pages in transit to include such content. Substitute certificates containing markings for WebMakerPlus originated from 16 distinct countries.

Manual Internet queries revealed that malware was responsible for an Issuer Common Name field value of "IopFailZeroAccessCreate." The certificates containing this value originated from 14 distinct countries. Disturbingly, each certificate contained the same 512-bit public key. This malware was also reported by [9].

It is somewhat surprising that these malware programs self-identify in the substitute certificates they generate, as an attacker can arbitrarily select values for the fields in a substitute certificate.

In addition to malware discoveries, we found that the names of two companies highly associated with spam were also present in numerous Issuer Organization fields. The names "Sweesh LTD", and "AtomPark Software Inc" were found in 39 and 20 substitute certificates, respectively. AtomPark offers tools for spammers including "email extractors" and "bulk mailers." Sweesh offers services to spammers to overcome "hurdles" faced by advertisers and publishers. Internet searches reveal that Sweesh may be responsible for the development of WebMakerPlus.

Not all of the root certificates found in the collected substitute chains were unique. In the 11,764 substitute chains 8,341 distinct roots were found. For example, 310 leaf certificates signed by "Fortinet" all used the same root certificate, and these were obtained from 155 distinct IP addresses. This behavior was consistent across many of the popular issuers identified (e.g., POSCO, Southern Company Services, Target Corporation). These organizations are likely using a single root to sign intermediate certificates and then deploying these at various endpoints where they operate TLS proxies.

5.2 Negligent Behavior

Where possible, we installed and characterized personal firewall software from many of the most common companies whose names were provided in the Issuer Organization, Issuer Organizational Unit, and Issuer Common Name fields of our collected certificates. We characterized the behavior of these solutions when running behind our own TLS proxy (setup using sslsplit and ARP poisoning) which issued certificates signed by an untrusted CA. While most solutions properly rejected our forged certificates, Kurupira, a parental filter that is responsible for 267 proxied connections in our dataset, did not. When visiting `google.com` and `gmail.com`, Kurupira replaced our untrusted certificate with a signed trusted one, thus allowing attackers to perform a transparent man-in-the-middle attack against Kurupira users without having to compromise root stores. In contrast, BitDefender not only blocked this forged certificate, but also blocked a forged certificate that resolved to a new root we installed.

We found TLS proxies that generate substitute certificates with weak cryptographic strength. Our original certificate has a public key size of 2048 bits. However, we found that 5,951 (50.59%) substitute certificates have public key sizes of 1024 bits and 21 certificates have public key sizes of 512 bits. In addition, 23 (0.20%) TLS proxies generated substitute certificates that used MD5

for signing, 21 (0.18%) which were also 512 bit keys. Interestingly, some TLS proxies generated certificates that have better cryptographic strength than our certificate. Seven (0.06%) used certificates with a key size of 2432 and five (0.04%) used SHA-256 for signing.

In addition to problems with cryptographic strength, we discovered that 49 (0.42%) substitute certificates claim to be signed by DigiCert, though none of them actually are. The original certificate from our secure web server is issued by DigiCert High Assurance CA-3, indicating the TLS proxy likely copied this field when creating the substitute. It is alarming that a TLS proxy would opt to copy this field, as it signifies a masquerading as the legitimate authority. It is possible that these proxies are operated by malicious individuals doing their best to not be detected by the user.

Finally, we note that 110 substitute certificates have modifications to the subject field. For 51 (0.43%) certificates, the subject did not match our website's domain. In many cases a wildcarded IP address was used that only designated the subnet of our website. In two cases the substitute certificate is issued to the wrong domain entirely: `mail.google.com` and `urs.microsoft.com`. These certificates appear to be legitimate for those domains and properly validate back to GeoTrust and Cybertrust roots, respectively.

6. RELATED WORK

The most closely related work in this field is a recent paper by Huang et al., which independently develops a measurement tool that is similar to ours and conducts a measurement study of TLS proxies that intercept the Facebook website [9]. Generally speaking, the advantage of Huang's methodology is that they may find proxies specifically targeting Facebook, whereas the advantage of our methodology is that we may detect proxies that intentionally whitelist a popular sites such as Facebook in order to avoid detection. A limitation of our method is that we cannot collect data from users with ad blockers. Thus results from Huang must be read with the postfix of "for facebook.com users," and results from our study must be read with the postfix of "for Internet users permitting ads."

In comparing our results to Huang, there are both similarities and differences. When inspecting the Issuer Field of substitute certificates, 13 of the 20, and 8 of the top 10 issuers also appear in Huang's measurements. Some of the bad actors we find (Sendori and IopFailZeroAccessCreate) were also found by Huang, in addition to some of the behavior by Kurupira. However, there are also some major differences between the results in these studies. The prevalence of proxies in our study is roughly twice what was measured by Huang (0.41% versus 0.20%). In addition, we find a wider array of malware, deceptive practices, and suspicious circumstances. Our measurements of WebMakerPlus represent malware found only in our study. Likewise, the presence of infections from Sweesh and AtomPark are unique to our study. We are the first to identify a parental filter replacing an untrusted certificate with a trusted one.

There are also some differences between the characteristics of the substitute certificates detected in our study and Huang. For instance, we find that chain depths of two or more certificates are more common. Chains with a depth of two or more certificates accounted for 20% of our substitute chains and 9% of Huang's. Note that the legitimate chains in both studies had a chain depth of two. In addition, 68 of our proxy results contained a chain depth of 5, compared to only 2 reported by Huang. Due to these depth differences, we also found more certificate chain sizes larger than 1000 bytes (20% vs 9%). We also see differences in the public key sizes of substitute certificates when comparing our results to those of Huang. In particular, we find less certificates using 512-bit key lengths (us: 21, Huang: 119) and the presence of keys larger than 2048 (us: 7, Huang: 0).

The only other paper to find evidence of TLS proxies is the work from The Netalyzer project, which analyzes the root store of Android devices [22]. Their primary findings include the use of manufacturer and vendor-specific certificates, the presence of unusual root certs, and third party apps that manipulate the root store. In addition, they find one case of a TLS proxy, out of 15,000 assessed TLS sessions. It is difficult to compare the prevalence (1 in 15K) to rates found by Huang and this paper because the sample is from users choosing to download the Netalyzer App.

A large body of work seeks to detect and prevent TLS proxies, generally regarding them as MitM attacks. Clark and van Oorschot [4] provide an extensive survey of this area, covering solutions that work with the CA system such as pinning [8], TACK [16], and Certificate Transparency [12, 21], as well as those that seek to validate self-signed certificates such as Convergence [15]. A lesser-known group of solutions acknowledges that there is an industry need for TLS inspection and some IETF drafts suggest notifying users when a proxy is present [18, 14] or even sharing session keys with proxies explicitly [19]. Our recent work surveying users indicates a strong pragmatic approach to TLS inspection, with willingness to allow this technique by employers, provided notification and consent is obtained [20].

7. CONCLUSION

Our work highlights the need for stronger protection and monitoring of the root store used by devices and browsers. The only way for TLS proxies to avoid a browser warning is if they generate substitute certificates that are valid; aside from compromising a certificate authority, both benevolent and malicious parties need to insert a certificate into the root store. Modifying the root store should require administrative privileges, and monitoring software should be used to remove certificates

from the store that are considered malicious or that are run by untrustworthy organizations.

In addition, better measurement tools are needed to understand the prevalence and nature of SSL proxies. The method used by Huang is still viable, but only works to detect proxies affecting a single server. Our measurements indicate that this undercounts proxies when that server is well-known, and yet measuring at well-known servers is the only way to get large amounts of data with this method. Using a Flash advertisement provides a scalable and robust method for detecting proxies, but this does not work if a user has an ad blocker installed. Moreover, we have found that most advertising networks no longer allow these types of advertisements. In the future, a community-driven, voluntary measurement platform would significantly help to collect these types of measurements.

8. ACKNOWLEDGMENTS

This work was supported by a 2014 Google Faculty Research Award, Sandia National Laboratories, and the National Science Foundation under Grant No. CNS-1528022. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

9. REFERENCES

[1] H. Adkins. An update on attempted man-in-the-middle attacks. http://googleonlinesecurity.blogspot.com/2011/08/update-on-attempted-man-in-middle.html.

[2] A. Barth, C. Jackson, and J. C. Mitchell. Robust defenses for cross-site request forgery. In *Proceedings of the 15th ACM Conference on Computer and Communications Security*, pages 75–88. ACM, 2008.

[3] T. Chiu. The growing need for SSL inspection. http://www.bluecoat.com/security/security-archive/2012-06-18/growing-need-ssl-inspection/, 2011. Accessed: 27 February, 2014.

[4] J. Clark and P. C. van Oorschot. SoK: SSL and HTTPS: Revisiting past challenges and evaluating certificate trust model enhancements. In *IEEE Symposium on Security and Privacy (SP)*, 2013.

[5] X. d. C. de Carnavalet and M. Mannan. Killed by proxy: Analyzing client-end tls interception software. In *Network and Distributed System Security Symposium (NDSS)*, 2016.

[6] Z. Durumeric, J. Kasten, M. Bailey, and J. A. Halderman. Analysis of the HTTPS certificate ecosystem. In *ACM Internet Measurement Conference (IMC)*, 2013.

[7] P. Eckersley. A syrian man-in-the-middle attack against facebook. https://www.eff.org/deeplinks/2011/05/syrian-man-middle-against-facebook, May 2011.

[8] C. Evans and C. Palmer. Certificate pinning extension for HSTS. http://tools.ietf.org/html/draft-evans-palmer-hsts-pinning-00. Accessed: 22 March, 2013.

[9] L. S. Huang, A. Rice, E. Ellingsen, and C. Jackson. Analyzing forged SSL certificates in the wild. In *IEEE Symposium on Security and Privacy (SP)*, 2014.

[10] G. Huston. Counting DNSSEC. https://labs.ripe.net/Members/gih/counting-dnssec. Accessed: 26 February, 2014.

[11] C. Jackson, A. Barth, A. Bortz, W. Shao, and D. Boneh. Protecting browsers from DNS rebinding attacks. *ACM Transactions on the Web (TWEB)*.

[12] B. Laurie, A. Langley, and E. Kasper. Certificate transparency, IETF RFC 6962. http://tools.ietf.org/html/rfc6962, Jun 2013.

[13] W. Lian, E. Rescorla, H. Shacham, and S. Savage. Measuring the practical impact of DNSSEC deployment. In *USENIX Security Symposium*, 2013.

[14] S. Loreto, J. Mattsson, R. Skog, H. Spaak, G. Gus, and M. Hafeez. Explicit trusted proxy in HTTP/2.0, Internet Draft. http://tools.ietf.org/html/draft-loreto-httpbis-trusted-proxy20-01, February 2014.

[15] M. Marlinspike. SSL and the future of authenticity. *Black Hat USA*, 2011.

[16] M. Marlinspike and T. Perrin. Trust assertions for certificate keys. http://tack.io/, 2013.

[17] MaxMind. Geolite. http://dev.maxmind.com/geoip/legacy/geolite/#IP_Geolocation. Accessed: 27 February, 2014.

[18] D. McGrew, D. Wing, Y. Nir, and P. Gladstone. TLS proxy server extension, Internet-Draft, TLS Working Group. http://tools.ietf.org/html/draft-mcgrew-tls-proxy-server-01, July 2012.

[19] Y. Nir. A method for sharing record protocol keys with a middlebox in TLS, Internet-Draft, TLS Working Group. http://tools.ietf.org/html/draft-nir-tls-keyshare-02, March 2012.

[20] S. Ruoti, M. O'Neill, D. Zappala, and K. Seamons. User attitudes toward the inspection of encrypted traffic. In *Symposium on Usable Privacy and Security (SOUPS)*, 2016.

[21] M. D. Ryan. Enhanced certificate transparency and end-to-end encrypted mail. In *Network and Distributed System Security Symposium (NDSS)*. Internet Society, 2014.

[22] N. Vallina-Rodriguez, J. Amann, C. Kreibich, N. Weaver, and V. Paxson. A tangled mass: The android root certificate stores. In *ACM Conference on Emerging Networking Experiments and Technologies (CoNEXT)*. ACM, 2014.