Investigating the Prevalence of Server Name Indication in the Current HTTPS Ecosystem

Shaown Sarker
NC State University
ssarker@ncsu.edu

Tae Hyun Kim
NC State University
tkim13@ncsu.edu

Douglas Reeves
NC State University
reeves@ncsu.edu

ABSTRACT

The original SSL protocol required that the server present its certificate without the knowledge of the domain name being contacted by the client. Due to this restriction, servers were prevented from supporting more than one domain per IP address. Because a server could only serve a single certificate per IP address, and each certificate could contain only a single Common Name. To overcome this two extensions to the X.509 certificate specifications and TLS protocol were developed - the Subject Alternate Names (SAN) list and the Server Name Indication (SNI). In this paper, we look into the support of the SNI extension to the TLS protocol in concurrent HTTPS websites. Our findings show that although SNI is a very capable solution for the single certificate per IP problem, most popular websites do not force the client to be SNI capable despite supporting SNI. We also show that websites with lesser traffic tend to have lesser support for SNI.

1. INTRODUCTION

Online, secure communication begins with end-to-end authentication. Secure Sockets Layer (SSL) and Transport Layer Security (TLS)[1] provides authentication for HTTPS traffic on the web. Coupled with a Public Key Infrastructure (PKI), SSL/TLS provides verifiable identities via certificate chains and private communication via encryption.

Key sharing in the web's PKI is facilitated by recent additions to the SSl protocol and extensions to X.509 certificates. An SSL certificate is a signed binding between a subject and a public key. Certificate Authorities (CA) issue valid certificates, who themselves have their own certificates. This creates a logical chain of certificates - starting from a *root* certificate through zero or more *intermediate* certificates, to a *leaf* certificate. In this chain, the certificate at a certain level is signed with the private key corresponding to the the certificate at the immediately previous level (with the sole exception of the root certificate, which is self-signed).

X.509 [4] is the most commonly used certificate management standard on the Internet. The certificates are commonly used as part of the SSL/TLS protocol (e.g., in HTTPS, IMAPS, etc). In SSL certificates, the subject is contained in the *Common Name* field; for leaf certificates, the *Common Name* is a domain name (e.g., `www.example.com`). SSL certificates also allow wildcard domains in the common name, so a certificate with a Common Name of *.example.com would cover both foo.example.com and bar.example.com. So, when a client contacts a server, it is essential that the domain name client is trying to contact is verified to be in the *Common Name* of the certificate. Otherwise, the client should reject the connection as this could be the result of a possible interception (e.g., man-in-the-middle attacks).

Due to the design of the original SSL protocol, the server was required to present the certificate without knowing which domain name the client was contacting. This effectively prevented servers from supporting more than one domain per IP address, as a server could only serve a single certificate per IP address, and each certificate could contain only a single *Common Name*. As a result, two extensions to the X.509 certificate specification and TLS protocol were developed:

SAN list - The Subject Alternate Names (SAN) extension allows a certificate to specify multiple alternate domain names to which the certificate should apply, effectively allowing a certificate to have multiple *Common Names*. For example, a certificate with a SAN list `[*.google.com, *.youtube.com]` would be accepted for both `www.google.com` and `m.youtube.com`.

SNI - The Server Name Indication extension to the TLS protocol allows a client to specify which domain it is trying to contact before the server presents its certificate. If both the client and the server support SNI, this allows the server to host SSL certificates for different domains on a single IP address; the server simply examines the SNI field to select which certificate it should send to the client.

Popular websites now commonly use hosting providers for distributing content and mitigating denial-of-service attack [3]. Websites are using third-party providers to host HTTPS content, but unfortunately, limitations of the TLS proto-

[1]Although SSL is a predecessor of TLS, both uses the same certificate architecture. When we refer to certificates or SSL certificates, it applies to both.

cols have made this challenging for hosting providers. TLS has historically assumed that a given IP address would be used to host only a single website's domains, and that therefor it would suffice for any given IP address to server a single certificate. To support multiple customers' HTTPS content, hosting providers use one or more of the following approaches:

- **One customer per IP address** - The most straightforward approach involves having a single customerâĂŹs certificate (possibly with multiple domains in a SAN list) allocated to any given IP address. This has the benefit of not requiring clients to support SNI, but comes at a high monetary cost, as IPv4 addresses have grown more scarce.

- **Cruise-liner certificates** - Finally, if a hosting provider can obtain custom certificates on behalf of its customers [6], it can craft certificates with SAN lists containing domains from multiple distinct customers. The problem with these "Cruise-liner" certificates is that the certificates cannot be modified piecemeal; adding or removing any customer's domain requires generating an entirely new certificate.

- **Multiple certificates per IP address** - Hosting providers can host multiple certificates on any given IP address by using SNI. This is less expensive than dedicating an IP address to a single customer, but unfortunately older clients like Internet Explorer on Windows XP and Android 2.x devices do not support SNI. Thus, hosting providers are often hesitant to implement a solution that leaves these clients unable to access customer websites.

  Although there have been research into overcoming this limitation of certificate on hosting providers using SAN list and cruise-liner certificate strategy [1], there has not been much exploration of the SNI adoption over the HTTPS ecosystem. In this paper, we look into the SNI support over most popular websites in the Internet. Our research questions in this paper are:

  RQ1 What is the current status of support for SNI over the most visited websites in the Internet?

  RQ2 Is there any relationship between the amount of web traffic and SNI support?

The rest of the paper is organized as follows: section 2 describes the methodology we used to answer our research questions, section 3 discusses our findings and the answers to the research questions posited here, section 4 describes the literature related to our work, and we conclude in section 5 by reiterating our findings.

2. METHODOLOGY

As more e-commerce sites emerge and more businesses begin to store or share sensitive documents online, the ability to host and extend secure sites is becoming increasingly important. In the old-fashioned versions, there were two issues that had to be addressed in order to host a secure site:

- **SSL Scalability**: In multi-user environments such as shared hosting, because the number of secure sites that can be hosted on a server is limited, the site density finally had to be low. [2]

- **IPv4 shortage**: Because the network endpoint can be identified only by the IP:Port binding, if the user wants to use the standard SSL port 443, each IP address must occasionally be assigned per user to host the secure site.

Also, Virtual hosting using SAN lists is both a fatal advantage and a weakness. They should use the same certificate and key within only one server. Therefore, SAN list does not provide any solution for the following situations:

- Because the domain owner is different, user can not obtain an integrated key for domains owned by others from CA.

- Even if the authority is delegated from the owner, the delegated administrator can misuse the issued key.

- Or they can delegate key management to each person for each domain.

SNI offers a more fundamental solution to these problems. Server Name Indication (SNI) is a TLS extension that can include virtual domains as part of the TLS negotiation process. The practical implication of this is that now you can use the virtual domain name or host name to identify the network endpoint. In addition, a new highly scalable Web Hosting repository to support SNI is provided. As a result, the security site density is much higher, and nevertheless only uses one IP address.

However, using SNI requires the client to support SNI. Most new browsers support SNI, but IE in Windows XP, and android 2.x doesn't support SNI. Based on these limitations, we categorize the support for SNI into following three categories to answer our research questions:

- **Support SNI** - The server is capable of servicing requests from SNI supporting clients and will perform the certificate exchange.

- **Requires SNI** - The server supports SNI, but requires that the client presents the domain name it is trying to contact beforehand, otherwise the server presents with a default certificate and will not perform the domain specific certificate exchange.

- **Forces SNI** - The server does not perform the certificate exchange at all without the client presenting the domain name it is trying to contact.

The second and third category are a subset of the first category. To determine whether the website server has SNI capabilities as categorized above, we used the open-source

[2]SSL Scalability. https://www.iis.net

tool openssl [3]. OpenSSL comes with a client tool that can connect to a secure server. The tool is similar to telnet or nc, in the sense that it handles the SSL/TLS layer but allows the user to fully control the layer that comes next.

To connect to a server, we need to supply a host-name and a port [8]. For example:

```
$ openssl s_client -connect www.feistyduck.com:443
```

To have SNI enabled in `s_client`, we need to use the `s_client` with the `-servername` switch [8]:

```
$ openssl s_client -connect www.feistyduck.com:443 -
servername www.feistyduck.com
```

Both of these command initiates the certificate exchange and we receive the certificate from the server. Based on the response from the website server from these two commands, we define our SNI support categories as follows:

- **Support SNI** - The server performs a certificate exchange with the `-servername` switch and we have a certificate from the server.
- **Requires SNI** - The server performs certificate exchange for both with and without the `-servername` switch, but the certificates are different, thus indicating that the server requires an SNI capable client for proper HTTPS communication.
- **Forces SNI** - The server performs a certificate exchange for only with `-servername` switch and we receive no certificate for a default connection.

To answer our research questions, we also retrieved the list of top 1 million websites [4] ranked by web-traffic published by Alexa [5]. Using the openssl tool, we collected the SNI support information fro the top 150,000 ranked websites along with the respective default and SNI certificates into a MySql database. Due to the geophysical location of the website host providers and their latency, we added a timeout of 10 seconds, so that if the certificate exchange does not take place between the 10 seconds from the start of the connect request, it is aborted.

3. RESULTS

Using our collected data we answer our research questions as follows:

RQ1 What is the current status of support for SNI over the most visited websites in the Internet?

We divided our data over rank ranges of 25,000 dividing into 6 ranges. Over these ranges, we measured the number of website server that supports, requires, and force SNI, which is depicted in figure 1. We can see that although SNI support is considerably high, a much smaller number of websites requires SNI and even a smaller number of websites actually forces SNI requirement upon clients. We can attribute this to the website host providers tendency to support SNI, but not denying service to legacy clients.

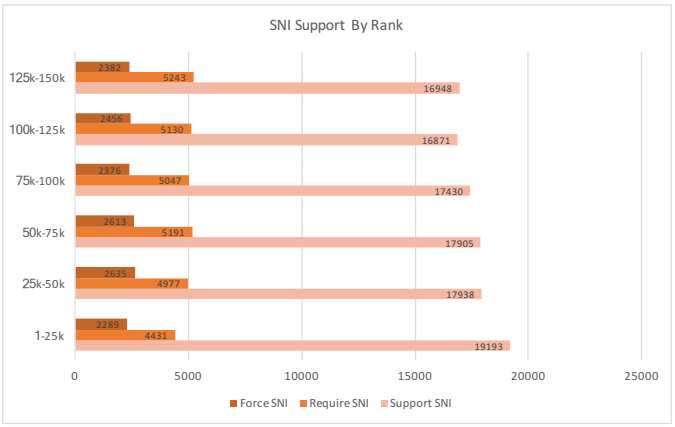

Figure 1: SNI Support by Ranks

RQ2 Is there any relationship between the amount of web traffic and SNI support?

To answer this question, we looked at the SNI support throughout the ranks in a cumulative manner, this is presented in table 1. From this data, we can see that none of the top 100 websites forces SNI upon the clients, and a very small number of websites requires SNI; although there is almost ubiquitous SNI support among the top 100 websites.

Table 1: SNI Support over Cumulative Ranks

Rank Range	Support SNI	Require SNI	Force SNI
<=100	97	25	0
<=1k	872	158	34
<=5k	4108	749	298
<=10k	8028	1658	774
<=20k	15525	3504	1786
<=50k	37131	9408	4924
Overall	106285	30019	14751

Figure 2 shows the data in table 1 as a percentage representation. From the data we can see that, as we move into lower ranked websites the support for SNI decreases, and at the same time we see a minute but steady increase in the require and force SNI categories. We can explain this observation in terms of website traffic. As the high volume websites desire to reach the wider set of clients, they tend not to force clients to have SNI capabilities, but lower volume traffic sites are often hosted by providers requiring or forcing SNI due to lowering of infrastructure and maintenance cost, as SNI enables the hosting provider to have multiple certificates per IP, but at the same time denying legacy clients of the server.

4. RELATED WORK

Cangialosi et al. in their 2016 work [1] looks into the private key sharing in the HTTPS ecosystem. They focus on the nature of key sharing between website hosting providers and the CDNs, certificate reissues and revocation as managed by

[3]https://www.openssl.org/
[4]http://s3.amazonaws.com/alexa-static/top-1m.csv.zip
[5]www.alexa.com

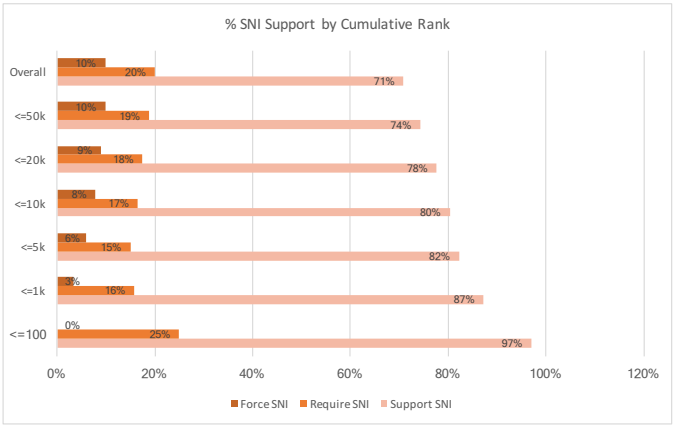

Figure 2: Percent of SNI Support by Cumulative Rank

third party providers, and the trust between organizations and their third party counterpart. In their work, the authors measure the number of certificates per IP based on the SAN list extension and cruise-liner certificate management through crafting custom certificates SAN lists. Our work extends on the work by Cangialosi et al. by exploring the other extension to TLS protocol - SNI and it's usage in the current HTTPS ecosystem.

Recent work by Liang et al. explores how Content Delivery Networks (CDN) manage SSL certificates when distributing HTTPS content to the customers [6]. They observe the distinction between custom and shared certificates, many instances of mismanagement, CAs neglecting to revoke certificates, and private key sharing. They specifically look into cruise-liner certificates. Our work compliments the work by Liang et al. by looking into the other prominent strategy available to overcome the shortcoming of a single certificate per IP.

There also have been numerous work on understanding and improving the SSL certificate ecosystem through measurements of CAs [2, 9, 5], and measurements of the cost of HTTPS security [7]. In contrast, our work builds upon these by focusing on the server level certificates and their exchange policies. We primarily focus on the issue of servers being limited to a single certificate per IP and how they tend to solve it through SNI while exploring the relationship between web-traffic and the SNI support.

5. CONCLUSION

In our work, we investigated the prevalence of SNI support among the top websites in the Internet. Our findings show that the websites with most web-traffic volume tend to support SNI, but they rarely require or force their clients to be SNI capable. On the other hand, websites with lower volume of traffic, either are reserved in their support of SNI or tend to require or force SNI capable clients for ease of infrastructure setup and maintenance cost. To reiterate our contributions, we collected and analyzed the current status of SNI support among the top 150,000 websites in the Internet along with their respective default and SNI enabled certificates.

References

[1] Cangialosi, F., Chung, T., Choffnes, D., Levin, D., Maggs, B. M., Mislove, A., and Wilson, C. Measurement and analysis of private key sharing in the https ecosystem. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security* (New York, NY, USA, 2016), CCS '16, ACM, pp. 628–640.

[2] Durumeric, Z., Kasten, J., Bailey, M., and Halderman, J. A. Analysis of the https certificate ecosystem. In *Proceedings of the 2013 Conference on Internet Measurement Conference* (New York, NY, USA, 2013), IMC '13, ACM, pp. 291–304.

[3] Holz, R., Braun, L., Kammenhuber, N., and Carle, G. The ssl landscape: A thorough analysis of the x.509 pki using active and passive measurements. In *Proceedings of the 2011 ACM SIGCOMM Conference on Internet Measurement Conference* (New York, NY, USA, 2011), IMC '11, ACM, pp. 427–444.

[4] Housley, R., Polk, W., Ford, W., and Solo, D. Internet x. 509 public key infrastructure certificate and certificate revocation list (crl) profile. Tech. rep., 2002.

[5] Laurie, B., Langley, A., and Kasper, E. Certificate transparency. Tech. rep., 2013.

[6] Liu, Y., Tome, W., Zhang, L., Choffnes, D., Levin, D., Maggs, B., Mislove, A., Schulman, A., and Wilson, C. An end-to-end measurement of certificate revocation in the web's pki. In *Proceedings of the 2015 Internet Measurement Conference* (New York, NY, USA, 2015), IMC '15, ACM, pp. 183–196.

[7] Naylor, D., Finamore, A., Leontiadis, I., Grunenberger, Y., Mellia, M., Munafò, M., Papagiannaki, K., and Steenkiste, P. The cost of the s in https. In *Proceedings of the 10th ACM International on Conference on emerging Networking Experiments and Technologies* (2014), ACM, pp. 133–140.

[8] Ristic, I. Openssl cookbook. *Feisty Duck* (2013).

[9] Zhang, L., Choffnes, D., Levin, D., Dumitras, T., Mislove, A., Schulman, A., and Wilson, C. Analysis of ssl certificate reissues and revocations in the wake of heartbleed. In *Proceedings of the 2014 Conference on Internet Measurement Conference* (New York, NY, USA, 2014), IMC '14, ACM, pp. 489–502.

Appendix

All of our code along with the data is available under the MIT license in the Github repository: `https://goo.gl/ybtNrj`