Efficiently Bypassing SNI-based HTTPS Filtering

Wazen M. Shbair,Thibault Cholez, Antoine Goichot, Isabelle Chrisment

Wazen M. Shbair
University of Lorraine, LORIA, France
shbair.wazen@loria.fr

IM2015 - Ottawa/Canada

12 May, 2015

Outline

HTTPS Traffic Dilemma

HTTPS Traffic

- Encryption is the commonly used solution to guarantee privacy and security.
- Based on Netcraft survey, 48% of websites moved to HTTPS with and increase of (+22%) since January 2013.

The Dilemma

- Content providers need SECURING contents over the Web.
- A local network administrator (from a company or university) that needs to monitor/filter access to some HTTPS websites.

The Main Issue

How can we monitor HTTPS traffic?

HTTPS Traffic Monitoring

Legacy solutions don't work

- Port Based, DNS and IP address are not reliable [1].
- Deep Packet Inspection (DPI): Encrypted traffic challenge [2].

Current solutions drawbacks

- Certificate Filtering: single certificate for multiple domains [3].
- HTTPS Proxy: It's hardly acceptable to trust third-party to screen sensitive information.

The Recent Method to Monitor HTTPS

Using Server Name Indication (SNI) extension for HTTPS traffic identification and filtering.

Overview of SNI

What is SNI ?

- SNI is an extension inside Client Hello Message.
- SNI originally proposed to support virtual hosting for websites use HTTPS [4].
- SNI helps the server for mapping between the requested domain from the client/browser and the corresponding SSL/TLS server certificate [5].

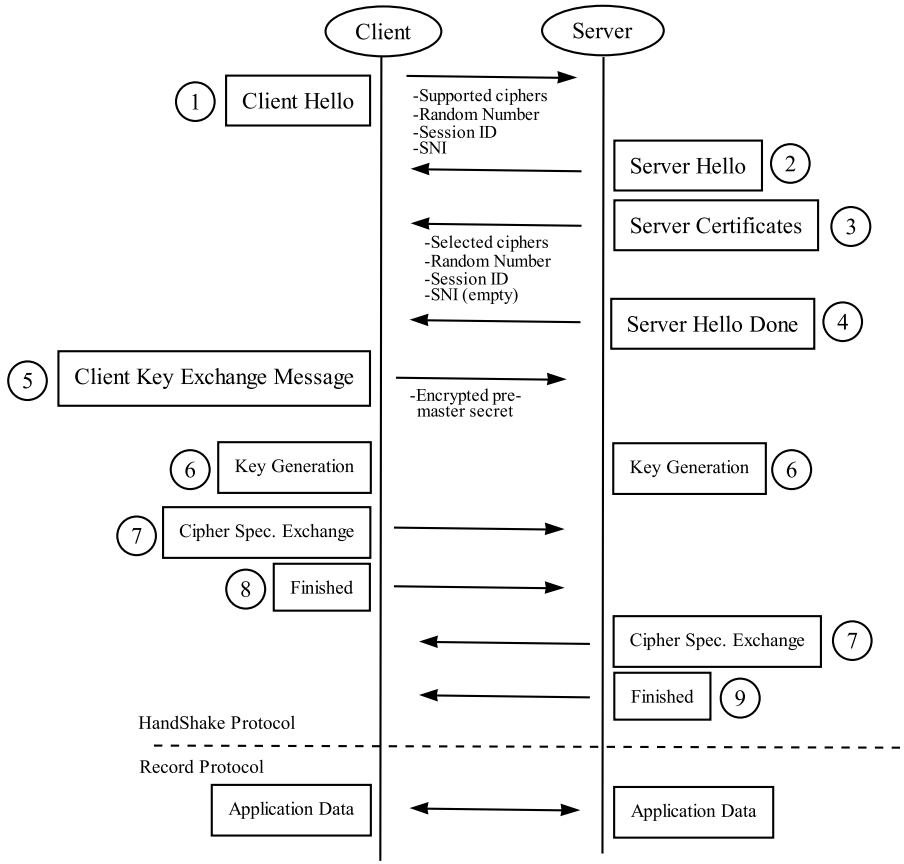

Figure : TLS Handshake

Overview of SNI

Why SNI ?

- SNI is a simply extracted value from the extensions list.
- Most of Web browsers and servers OS support SNI [4]
- The user's privacy is untouched.
- Implemented in many firewalls system such as Sphirewall and SOPHOS.

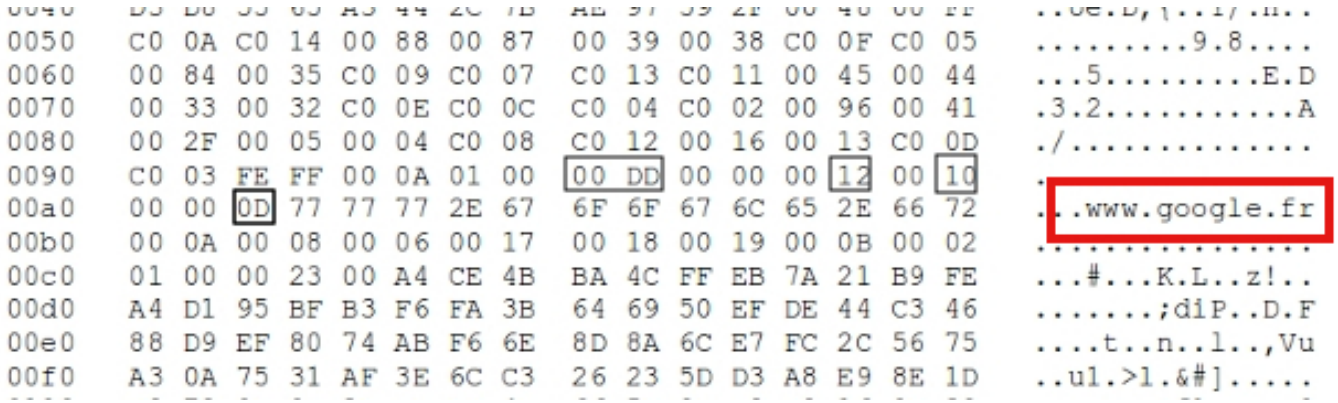

Figure : Server-name in SNI Extension

SNI-Based HTTPS Filtering

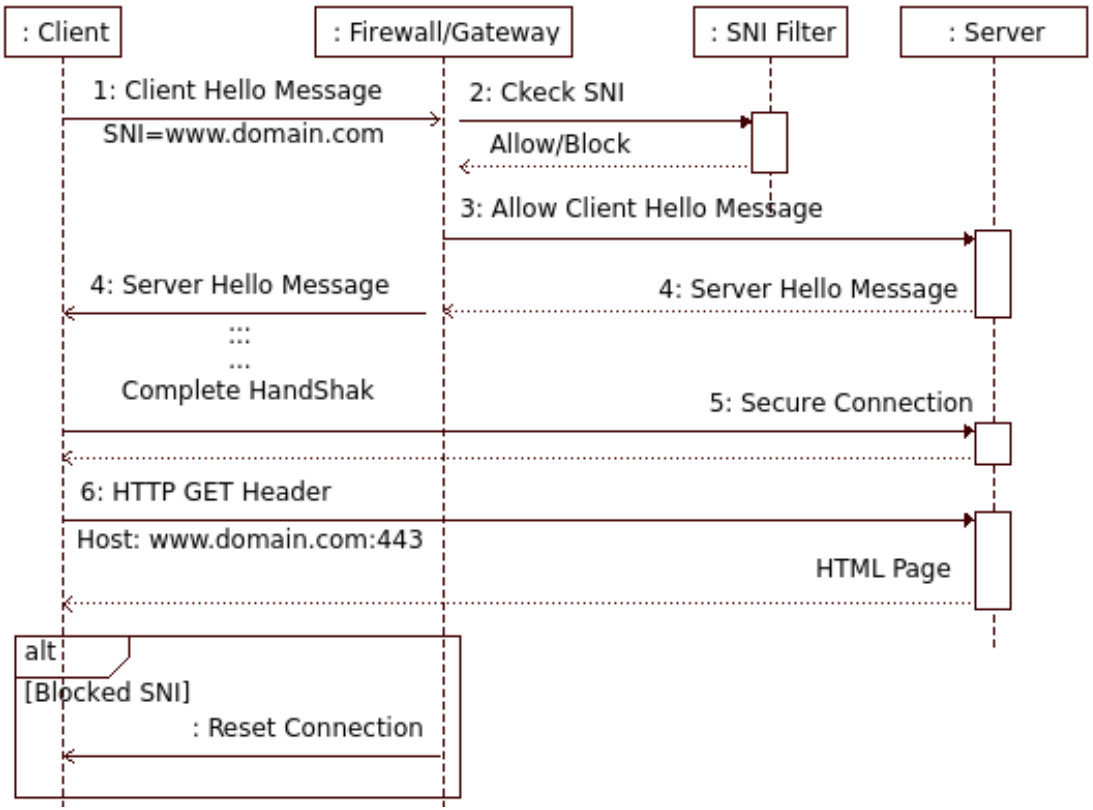

Figure : SNI Filtering Sequence Diagram

SNI-Based HTTPS Filtering

What about the reliability of SNI-Filtering ?

The usage of SNI for identifying the source of HTTPS traffic needs to be evaluated.

Strategies Exploiting SNI-Filtering Weakness

1. Backward Compatibility

Based on RFC6066, meaning that TLS clients that support the extensions can talk to TLS servers that do not support the extensions, and vice versa [5].

2. Shared Server Certificate

The alternative name field in the certificate standard X.509 makes it possible to hold a set of domain names using the same certificate [3].

Bypassing Firewall Systems

These weaknesses can be used for circumventing firewalls relying on SNI to monitor and filter HTTPS traffic.

1. Backward Compatibility

Backward compatibility can be used to cheat firewalls as follow:

1. Remove SNI extension form the client-hello.
2. Insert an alternative/faked "server-name" in the SNI.

Example: assume Facebook is blocked

- Create TLS Java Socket :
  `TARGET_HTTPS_SERVER=facebook.com`
  `TARGET_HTTPS_PORT=443`
- Create SNI Object with `server_name=bypassf@ceb00k.com`
- Send HTTP host header with real address of the blocked website:
  `GET/HTTP/1.1/r/n`
  `HOST:facebook.com:443`

2. Shared Server Certificate

This can be used to get access to a banned website by sending the SNI for non-banned websites sharing the same server certificate

Example: assume Youtube is blocked

- Create TLS Java Socket :
  `TARGET_HTTPS_SERVER=maps.google.com`
  `TARGET_HTTPS_PORT=443`
- Create SNI Object with `server_name=maps.google.com`
- Send HTTP host header with real address of the blocked website:
  `GET/HTTP/1.1/r/n`
  `HOST:youtube.com:443`


Escape
BYPASSING HTTPS FILTERING

Implementation of a Web Browser Plug-in

Escape

- An add-on named Escape[1] developed for the Firefox web browser.
- The motivation is the strong relation between our work and web browsing.
- Escape is the technical solution we chose to get the control over the TLS handshake.

How it works?

It works as a **LOCAL** MITM that intercepts the web browser requests regarding TLS and creates its own TLS connection with both sides.

[1] http://madynes.loria.fr/Research/Software

Implementation of a Web Browser Plug-in

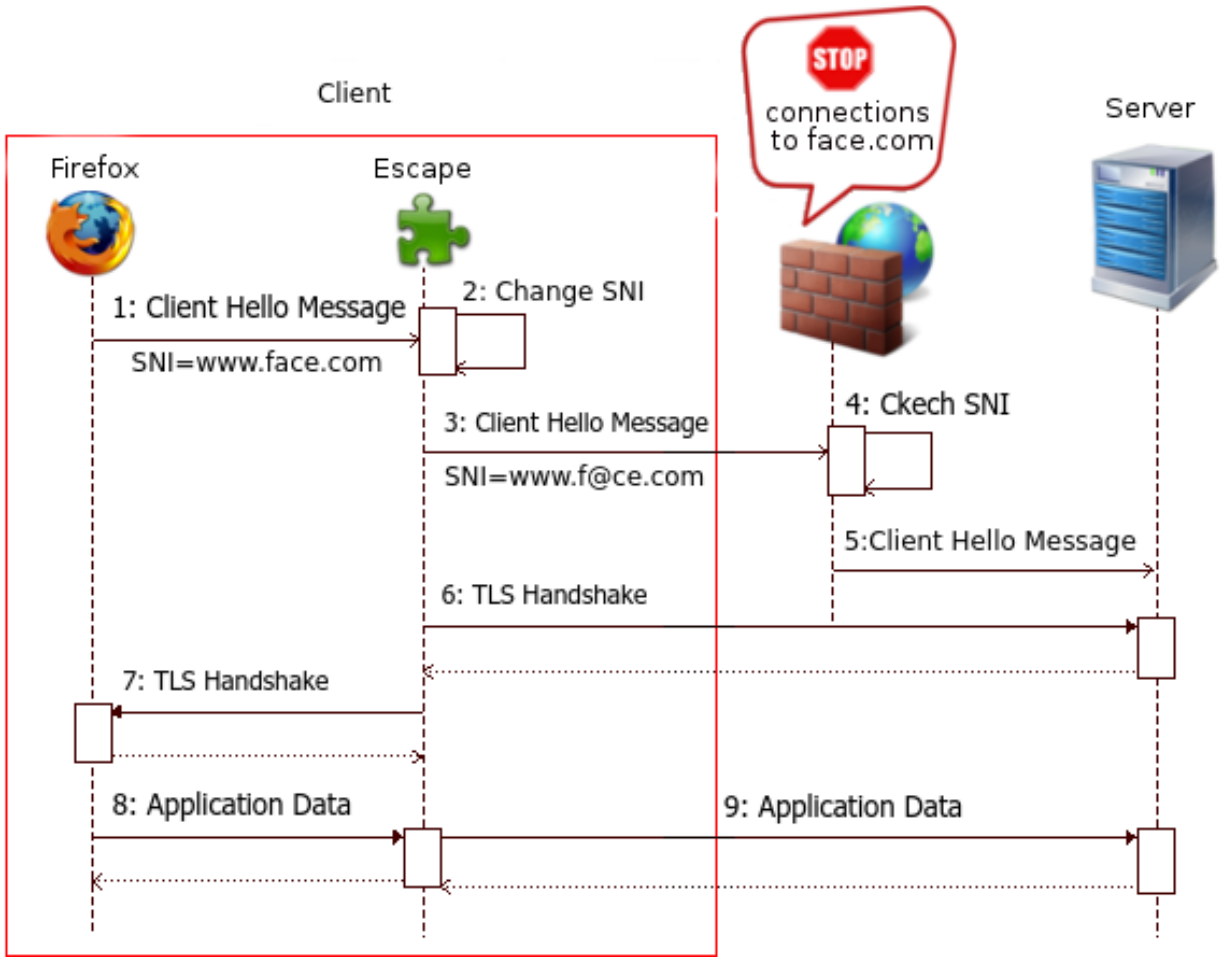

Figure : Escape plug-in interaction

Evaluation

Evaluation against SNI-based firewalls

- Firewall systems with HTTPS filtering based on SNI:
  - Sphirewall (0.9.9.27)
  - IPFire (2.15)
  - Untangle NG Firewall (10.2.1)
- Some tested websites: Google Search, Facebook,Youtube, Twitter, Linkedin, Blogger, Netflix, Dropbox, ..etc.

Results

- All tested web servers implement backward compatibility following RFC6066.
- We successfully tested the plug-in against 3 firewalls and Top 20 visited websites.

Conclusion and Future Works

Conclusion

- The amount of HTTPS website increasing rapidly.
- SNI-Filtering has two weaknesses, regarding the backward compatibility and multiple services using a single certificate.
- The "Escape" plug-in is our proof of concept exploiting SNI weaknesses.

Future Works

- Develop a new generation of monitoring tools and methods using advanced machine learning techniques to classify HTTPS traffic.

References

[1] S. Valenti, D. Rossi, A. Dainotti, A. Pescap, A. Finamore, and M. Mellia, "Reviewing traffic classification," in *Data Traffic Monitoring and Analysis* (E. Biersack, C. Callegari, and M. Matijasevic, eds.), vol. 7754 of *Lecture Notes in Computer Science*, pp. 123–147, Springer Berlin Heidelberg, 2013.

[2] Y. Xue, D. Wang, and L. Zhang, "Traffic classification: Issues and challenges," in *2013 International Conference on Computing, Networking and Communications (ICNC)*, pp. 545–549, 2013.

[3] 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*, pp. 511–525, IEEE, 2013.

[4] L. Vlker, M. Noe, O. P. Waldhorst, C. Werle, and C. Sorge, "Can internet users protect themselves? challenges and techniques of automated protection of HTTP communication," in *Computer Communications*, vol. 34, pp. 457–467, 2011.

[5] D. Eastlake, "RFC 6066 - the transport layer security (TLS) extensions: Extension definitions," 2011.