Encrypted SNI

No Name

September 20, 2019

1 Introduction

In this note we try formalize the privacy goals of Encrypted SNI [1].

2 Notation

TLS is an authenticated key exchange protocol composed of sub-protocols. Importantly, TLS has a handshake and record protocol. The handshake protocol uses handshake messages to perform key exchange and its many facets, including, among other things, cryptographic algorithm selection and peer authentication. Let $h \in \{0,1\}^{2^{16}}$ be a handshake message composed of arbitrary data, and let $\mathbf{H}$ be the set of possible handshake messages.

The record protocol uses messages with headers consisting of type, length, and version to send data between peers. This data is either a plaintext handshake message or encrypted blob. For our analysis, let $r = (d, p)$ be a record composed of direction $d \in \{0,1\}$ and payload $p \in \{0,1\}^{2^{16}}$. Let $\mathbf{R}$ be the set of all possible records. Let $\mathbf{P} = \{0,1\}^{2^{16}}$ be the set of possible payloads. Note, by definition $\mathbf{H} \subset \mathbf{P}$. Let $\mathsf{Message}$ be a function that returns the plaintext message for a given record r, or nothing if the record is encrypted.

We model a TLS handshake as a trace of records and their metadata sent between a client and server. (Metadata may include, among other things, record lengths as observed on the wire.) Formally, let $\vec{t} =< r_0, r_1, \ldots, r_n$ be a trace of n records, where $r_i = (d_i, p_i) \in \mathbf{R}$. Let $\mathbf{T}$ be a set of traces.

Without loss of generality, in TLS 1.3, there are three types of messages observed on the wire: ClientHello (CH), ServerHello (SH), and ApplicationData (AD). Let $\mathbf{CH}$ and $\mathbf{SH}$ be the set of possible CH and SH handshake messages generated by clients and servers, respectively. Each CH, SH, and AD message belongs to $\mathbf{P}$.

Let $N \in \{0,1\}^{2^{16}-1}$ be a name, and let $\mathbf{N}$ be the set of names. CH messages typically carry names as one of their parameters.

Let $\mathsf{ClientConfig}$ be a set of parameters that determine a client's configuration, including supported ciphersuites, named groups, extensions, etc. Let $\mathbf{CC}$ be the set of all client configuration parameters. Similarly, let $\mathsf{ServerConfig}$ be a set of parameters that determine a server's configuration, and let $\mathbf{SC}$ be the set of all client configuration parameters. Clients produce CH messages to a given server, identified by its name N, and configuration $\mathsf{ClientConfig}$, using a function $\mathsf{GenerateCH} : \mathbf{N} \times \mathbf{CC}$. Servers produce SH messages (and do other parameter selection) based on a CH message and server configuration, using a function $\mathsf{GenerateSH} : \mathbf{CH} \times \mathbf{SC}$.

The function $\mathsf{Metadata}$ takes as input a CH or SH and returns the set of metadata, or parameters, associated with the message, including the server name N, ciphersuite list, named groups, and other extensions that are visible on the wire. The function $\mathsf{PublicMetadata}$ returns the same output as $\mathsf{Metadata}$, except that the server name is omitted.

Using the notation above, every handshake is a probabilistic function of some server name N, $\mathsf{ClientConfig}$, and $\mathsf{ServerConfig}$. In particular:

- A client generates a CH message using N and $\mathsf{ClientConfig}$.
- The recipient server generates a SH message using the input CH and $\mathsf{ServerConfig}$.
- The remainder of the handshake contains encrypted AD messages.

Thus, let $\mathsf{Handshake}$ be a function that takes as input N, $cc = \mathsf{ClientConfig}$, and $sc = \mathsf{ServerConfig}$ and produces a trace $t = r_1, r_2, r_3, \ldots,$ where $\mathsf{Message}(r_1)$ is a CH and $\mathsf{Message}(r_2)$ is a SH. Let $\mathsf{PublicHandshake}$ be a similar function that computes $t = r_1, r_2, r_3, \cdots = \mathsf{Handshake}(t, cc, sc)$ and returns $t^v = \mathsf{PublicMetadata}(r_1), \mathsf{PublicMetadata}(r_2), r_3, \ldots$. That is, it returns publicly visible metadata associated from a handshake trace.

Finally, we define a function SNI which takes as input a handshake trace t and returns the name N which was used to generate t.

3 Security Model

In this section, we describe two variants of an ESNI adversary: passive and active. We derive definitions from the classic "find-and-choose" notion of indistinguishability. Informally, in each game, the adversary Adv is given information about a game and must present two options for the challenger. The challenger chooses one option at random and presents the result of some operation applied to the selected option to Adv, who must then identify which of the two options was selected. (This is typically the encryption of one message.) If the Adv's advantage in making this selection is negligibly later than 0.5, then the options are indistinguishable under the challenger's operation.

In porting this definition to ESNI, we seek to capture indistinguishability of handshake traces derived from names. That is, the challenger's operation is to generate a public handshake trace based on a randomly chosen name. Specifically, Adv chooses two names N_0 and N_1 during a "find" phase, and presents them to the challenger C. C then generates $t_b^v = \mathsf{PublicHandshake}(N_b)$, where $b \leftarrow \{0, 1\}$, and presents this to Adv. The adversary then presents its

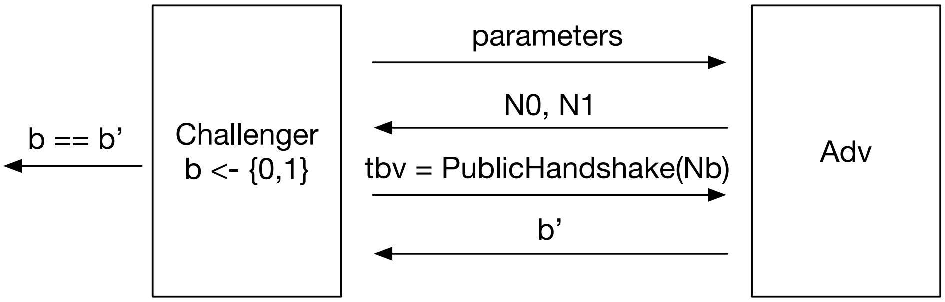

Figure 1: Passive ESNI Privacy game

selection b' to the challenger. The output of the game depends on whether or not $b = b'$. This is shown in Figure 1.

We capture this definition in the following NameGame, which is defined in Algorithm 1.

Algorithm 1 NameGame

```
1: On input security parameter $\lambda$, $\mathbf{N}$, $\mathbf{CC}$, $\mathbf{SC}$, and adversary Adv a challenger $C$ initializes Adv with $\mathbf{CC}$, $\mathbf{SC}$, and N.
2: Adv presents the challenger with two distinct names $N_0$ and $N_1$, where $N_0 \neq N_1$.
3: $C$ chooses a random bit $b$ and returns $t_b^v = \mathsf{PublicHandshake}(N_b)$ to Adv.
4: Adv replies with a bit $b'$.
5: Output 1 if $b = b'$, otherwise output 0.
```

We say that Adv wins NameGame if it outputs 1, i.e., if Adv was able to determine distinguish two handshakes using the same ESNI (or not). We define the advantage Adv has in this game as $|\Pr[\mathsf{NameGame}(\lambda, \mathbf{N}, \mathbf{CC}, \mathbf{SC}, \mathsf{Adv})] - \frac{1}{2}|$. That is, the probability that Adv guesses b correctly less the probability that it guesses incorrectly.

We say that a handshake trace is *SNI agnostic* if, for all PPT adversaries Adv, Adv's advantage in winning NameGame is negligibly small in λ.

3.1 Active Adversary

An active adversary has the ability to generate handshake traces at will. We assume servers are neither malicious nor compromised. Therefore, adapting our model for privacy requires extending NameGame to give Adv an oracle for producing handshake traces with an SNI of its choosing. We define this oracle as $\mathcal{O}_H$, and it takes as input an SNI $N \in \mathbf{N}$ and returns a handshake trace $H(N) \in \mathbf{H}$. The modified game, called ActiveNameGame, proceeds as follows:

As before, we say that Adv wins ActiveESNIGame if it outputs 1. We define the advantage Adv has in this game as $|\Pr[\mathsf{ActiveESNIGame}(\lambda, \mathbf{N}, \mathsf{config}, \mathsf{Adv})] -$

Algorithm 2 ActiveNameGame

```
On input security parameter $\lambda$, $\mathbf{N}$, $\mathbf{CC}$, $\mathbf{SC}$, and adversary $\mathsf{Adv}$ a challenger $C$ initializes $\mathsf{Adv}$ with $\mathbf{CC}$, $\mathbf{SC}$, $\mathsf{N}$, and access to $\mathcal{O}_H$.
$\mathsf{Adv}$ queries $\mathcal{O}_H$ with names values of its choosing from $\mathbf{N}$ at most $\mathsf{poly}(\lambda)$ times.
$\mathsf{Adv}$ presents the challenger with two distinct names $N_0$ and $N_1$, where $N_0 \neq N_1$.
$C$ chooses a random bit $b$ and returns $t_b^v = \mathsf{PublicHandshake}(N_b)$ to $\mathsf{Adv}$.
$\mathsf{Adv}$ continues querying $\mathcal{O}_H$ at most $\mathsf{poly}(\lambda)$ times.
$\mathsf{Adv}$ replies with a bit $b'$.
Output 1 if $b = b'$, otherwise output 0.
```

$\frac{1}{2}|$. And we say that a handshake trace is *SNI agnostic* with respect to an active attacker if Adv's advantage in winning $\mathsf{ActiveESNIGame}$ is negligibly small in λ.

References

[1] Eric Rescorla, Kazuho Oku, Nick Sullivan, and Christopher A. Wood. Encrypted Server Name Indication for TLS 1.3. Internet-Draft draft-ietf-tls-esni-04, Internet Engineering Task Force, July 2019. Work in Progress.