Network Security II

Question 1 ***DNS Walkthrough***

Your computer sends a DNS request for "www.google.com"

(a) Assume the DNS resolver receives back the following reply:

```
com. NS a.gtld-servers.net
a.gtld-servers.net A 192.5.6.30
```

Describe what this reply means and where the DNS resolver would look next.

(b) If an off-path adversary wants to poison the DNS cache, what values does the adversary need to guess?

(c) What are some issues with using TLS to secure DNS?

Question 2 ***TLS threats***

An attacker is trying to attack the company Boogle and its users. Assume that users always visit Boogle's website with an HTTPS connection, using ephemeral Diffie-Hellman. You should also assume that Boogle does not use certificate pinning. The attacker may have one of three possible goals:

1. Impersonate the Boogle web server to a user
2. Discover some of the plaintext of data sent during a past connection between a user and Boogle's website
3. Replay data that a user previously sent to the Boogle server over a prior HTTPS connection

For each of the following scenarios, describe if and how the attacker can achieve each goal.

(a) The attacker obtains a copy of Boogle's certificate.

(b) The attacker obtains the private key of a certificate authority trusted by users of Boogle.

(c) The attacker obtains the private key corresponding to an old certificate used by Boogle's server during a past connection between a victim and Boogle's server. Assume that this old certificate has been revoked and is no longer valid. Note that the attacker does not have the private key corresponding to current certificate.

Question 3 ***TLS protocol details***

Depicted below is a typical instance of a TLS handshake.

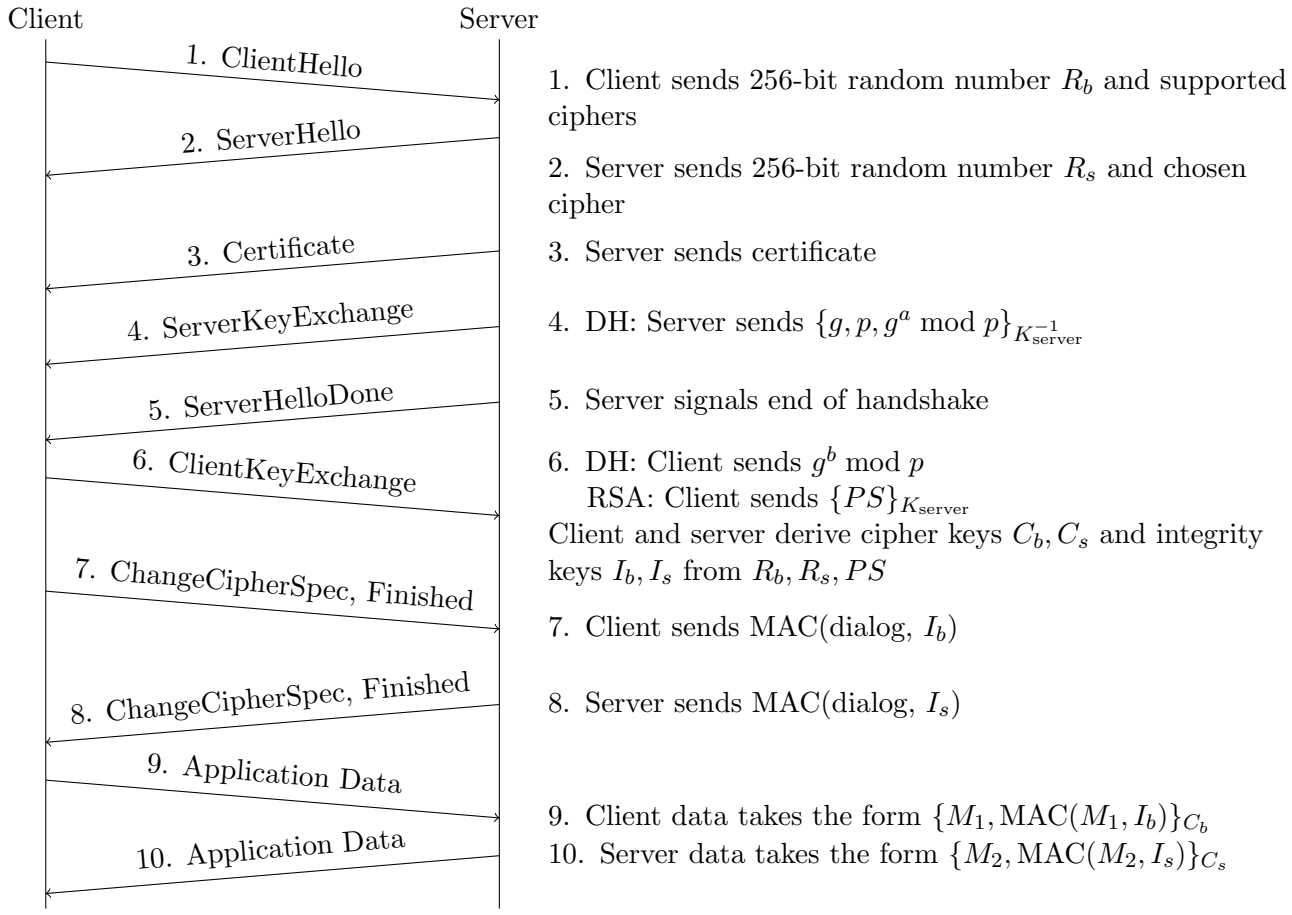

Figure 1: TLS 1.2 Key Exchange

(a) What is the purpose of the *client random* and *server random* fields?

(b) ClientHello and ServerHello are not encrypted or authenticated. Explain why a man-in-the-middle cannot exploit this. (Consider both the Diffie-Hellman and RSA case.)

(c) Note that in the TLS protocol presented above, there are two cipher keys C_b and C_s. One key is used only by the client, and the other is used only by the server. Likewise, there are two integrity keys I_b and I_s. Alice proposes that both the server and the client should simply share one cipher key C and one integrity key I. Why might this be a bad idea?

(d) The protocol given above is a simplified form of what actually happens. After step 8 (ChangeCipherSpec), the protocol as described above is still vulnerable. What is the vulnerability and how could you fix this?