CS255 Problem Set 3

Andrew Moreland – andymo@stanford.edu – 005752336
Worked with Atticus Christensen

March 12, 2014

---

1. The modified protocol is not secure. This is as anyone, notably any attacker, can compute SHA-256(ℓ), so if someone were to impersonate the client, they could pass the challenge and response by computing SHA-256(ℓ), where ℓ is the challenge. Therefore, this modified protocol is not secure.

2. We do not need to check the issuer field because we have pinned our root CA. In other words, if the certificate we receive from the server is valid it must have been signed by our pinned root CA. This implies that the certificate is issued by the root CA, so checking the issued by field is redundant.

3. Assume that the attacker has access to the storage location of the certificates used by the client.

   If, instead of storing the certificate and deriving a secret key from it and a password, you had simply stored the derived secret key on disk, then the attacker would be able to mimic the client by simply deserializing that secret key and using it to sign the challenge.

   Similarly, if no password were used (as is often the case with SSH keys), then the attacker with access to the certificate would be able to derive the secret key and mimic the client.

4. (a) Symmetric-key challenge-response is faster because it requires fewer communications. In particular, it requires one from the client to the server, rather than the two that assymetric challenge-response requires: one from the server to the client and then a signature response.

   (b) If the server has a small amount of trusted storage and a large number of clients then (by the pigeonhole principle) it cannot store unique secret keys for all clients. This means that it must reuse keys in which case clients can impersonate eachother.

5. (a) The server would simply store the (salted) hashed password of the client. It is unsafe to simply store the client's password because if the server is compromised then the client's password is leaked. Moreover, we can't just store a hash of the client's password because this hash is possibly vulnerable to rainbow table cracking attacks.

   In order to authenticate, the client would transmit their password which would then be salted and hashed and compared to the stored value.

   (b) If the attacker is able to trick the client into believing that it is actually the server, then the client will submit its password using the hacker's public key, so the attacker is able to learn the password. Even after the certificate expires, the hacker will know the user's password and therefore be able to authenticate with the real server as the user.