Cryptography

Lab 5 – 10 VI

Each Project: 40 points. You need both: present an implementation and provide a short report describing: problem statement, proposed solution/results.

Each project can be prepared by up to 2 students.

1. Implement attack described in [3].
2. Modify TLS library in such a way that the random values used in TLS handshake are predictable to you. Then write a script which given a (*e.g.,* use tdpdump) transcript of the TLS session decrypts it for you.
3. Implement dining-cryptographers protocol.
4. Implement a mobile app which has a functionality similar to "Tinderella" (`www.youtube.com/watch?v=bLoRPielarA`). The goal of the protocol performed between users P_i and P_2 with inputs x_1, x_2 respectively is to compute $F_i(x_i, x_2) = F_2(x_1, x_2) = b$ where $b = x_1 \wedge x_2$.

   Your app should not let anyone to learn about user's inputs (you can use Yao's garbled circuits or GSW protocol).
5. Write a mobile app which lets two users to tell if they are nearby. Implemented functionality returns 1 if two users are within a given distance and returns 0 in any other case.
6. Implement a command-line Helios voting [1] client.

   Your implementation should output:

   - an encryption of a ballot,
   - (together with) zero-knowledge proof of knowledge of exponent (randomness used).
   - commitment to randomness.

   Follow technical description which you can find on the website [2]. The output of your program needs to be in json format and be compatible with the Helios implementation *i.e.,* one should be able to use Helios verifier (`https://github.com/benadida/helios-server/blob/master/heliosverifier/verify.html`) to audit correctness of your ballot encryption.
7. Implement a mobile app which lets one to verify correctness of ecnrypions in Helios voting system. The verifier takes as an input an encrypted ballot (in json format) and checks correctness of zero-knowledge proofs attached to it.
8. Prepare a mobile app(s) which implements Feige-Fiat-Shamir identification protocol. Such an app can be used then as user's digital ID.

References

[1] Ben Adida. Helios: Web-based open-audit voting. In *USENIX Security Symposium*, volume 17, pages 335–348, 2008.

[2] Ben Adida. http://documentation.heliosvoting.org/verification-specs/helios-v4, 2012.

[3] Eran Tromer, Dag Arne Osvik, and Adi Shamir. Efficient cache attacks on aes, and countermeasures. *Journal of Cryptology*, 23(1):37–71, 2010.