NYU, cs6903/4783, Project 2.7 (Designing an end-to-end cryptography solution to protect your data application from attacks):

The project consists of a software implementation and demonstration of your method to enhance "your (ideally, real-life) data processing application" so that it satisfies end-to-end security against attacks that we have studied in this class, or enhanced versions of them, including at least the following attacks: data eavesdropping, data modification, data originator spoofing and data replay. End-to-end security should hold across the entire communication path from the sending device to the receiving device, including the link connecting the device to the data processing application. Your project should consist of the following steps:

1) Choose your data processing application; suggestions include the following:

1. Party1 posts messages and/or files on a social network that (some or all) other Parties can read,
2. Party1 posts data on other public communication channels (e.g., a web page, a blog) that (some or all) other Parties can read,
3. Party1 stores data files on a cloud server and later allows Party 1 (or other Parties) to retrieve (some or all of) them,
4. your favorite web-based data processing application (if approved by the instructor), or
5. your favorite smartphone-based data processing application (if approved by the instructor).

2) Analyze how attacks studied in this class impact the end-to-end security, functionality, and/or efficiency of your data processing application; in other words, what can go wrong as a consequence of one or more of these attacks. You should start with attacks like data eavesdropping, data modification, data originator spoofing and data replay from a man-in-the-middle, across the entire communication path from the sending device to the receiving device, including the link connecting the device to the data processing application. Then you should expand your analysis as much as possible, including, where applicable, variations/combinations of the mentioned attacks, and/or attacks carried out by other parties involved in your application (e.g., service providers).

3) Design methods to protect your data processing application against as many of these attacks as possible. Your methods should be based on cryptographic primitives studied in our class (up to Lecture 7).

4) Implement your cryptography methods in any programming language (preferably, C, C++, Rust or Python), based as much as possible on open-source software libraries (e.g., OpenSSL, Crypto++, Miracl, etc.), and combined as much as possible with a real-life data processing application (e.g., a real-life social network, cloud storage, web/smartphone application). Unleash your creativity in performing this combination, with the understanding that any combination will be acceptable (including performing the cryptography on your application and then using the result on the real-life application). Implement your non-cryptography methods, if any, using your favorite programming language. You are not required to use software from your chosen real-life data processing application (if you do, that's essentially considered extra credit). Implementing

your own data processing application is considered out-of-scope, unless your justification for doing that is exceptionally accepted by the instructor. Your methods should keep the requirements of

- data processing functionality (that is, your method should preserve your data processing application's functionality);
- data confidentiality,
- data integrity attack detection,
- data originator spoofing detection
- data replay attack detection, and
- efficiency.

Before starting implementation, you have the (strongly encouraged) option to check your design with the instructor, who will not go over too many details of your design but will tell you if he can see any major design flaws or omissions. The instructor may ignore such requests if obtained too close to the deadline.

5) Prepare a project presentation file (using, for instance, Microsoft Powerpoint). You must include in your presentation a detailed description of above steps 1-4, together with a demonstration (using videos or screenshots) of how your methods work in conjunction with the specific real-life data processing application. In your demonstration, you might likely need to run the real-life data processing application. You will use this presentation file for your in-person or recorded 10/15-min workshop presentation.

Your submission will be judged based on the following project grading criteria:

1. Application choice (i.e., if you chose a real-life data processing application, how interesting is the application, etc.)
2. Security analysis (i.e., if you analyzed all or at least the most important attacks to the chosen data processing application, etc.)
3. Design validity (i.e., if you choose appropriate cryptographic primitives, if the schemes instantiating the primitives and their key length parameters are valid choices in terms of security and efficiency)
4. Implementation validity (i.e., if your software, after inspection of the presentation demonstration and some amount of testing, seems to satisfy correctness; if your software is easy to use / run, has a well-written readme file, etc.)
5. Demonstration/presentation quality (i.e., if the presentation is well written and insightful, if the demonstration is clear and insightful, etc.).