CS 1653: Applied Cryptography and Network Security
Spring 2011

Term Project, Phase 4

Assigned: Tuesday, March 15 **Due:** Tuesday, April 5 11:59 PM

---

1 Background

In this phase of the project, you will further extend your file sharing applications to protect against a few more classes of security vulnerabilities. You will be provided with a threat model describing the types of assumptions that you will be able to make regarding the principals in the system, and will be given a list of specific classes of threats for which your system must provide protections. Your deliverables for this phase of the project will again include (i) a brief writeup describing your proposed protections for each type of threat, as well as a description of why these protections are sufficient, and (ii) a set of modified file sharing applications that implement each of your protections.

2 Trust Model

In this phase of the project, we are going to focus on implementing another set of the *security features* required of our trustworthy file sharing service. Prior to describing the specific threats for which you must provide protections, we now characterize the behavior of the four classes of principals that may be present in our system:

- **Group Server.** The group server is entirely trustworthy. In this phase of the project, this means that the group server will only issue tokens to *properly authenticated* clients and will properly enforce the constraints on group creation/deletion/management specified in previous phases of the project. The group server is *not* assumed to share secrets with the file servers in the system.
- **File Servers.** In this phase of the project, file servers will be assumed to be largely untrusted. In particular, file servers might leak files to unauthorized users or steal user tokens.
- **Clients.** We will assume that clients are not trustworthy. Specifically, clients may attempt to obtain tokens that belong to other users and/or modify the tokens issued to them by the group server to acquire additional permissions.
- **Other Principals.** You should assume that *all* communications in the system might be intercepted by a *active attacker* that can insert, reorder, replay, or modify messages.

3 What do I need to do?

Given the above trust model, we must now consider certain classes of threats that were not addressed in the last phase of the project. We now describe the threats that your group must address, as well as requirements for your implementation and writeup.

3.1 Threats

In this phase of the project, your group must develop defenses against the following classes of threats:

T1. Message Reorder, Replay, or Modification. After connecting to a properly authenticated group server or file server, the messages sent between the user and the server might be reordered, saved for later replay, or otherwise modified by an active attacker. You must provide users and servers with a means of detecting message tampering, reordering, or replay. Upon detecting one of these exceptional conditions, it is permissible to terminate the client/server connection.

T2. File Leakage. Since file servers are untrusted, files may be leaked from the server to unauthorized principals. You must develop a mechanism for ensuring that files leaked from the server are only readable by members of the appropriate group. As in previous phases of the project, we stress that the group server cannot be expected to know about all file servers that are in existence. Further, your proposed mechanism *must* ensure that some level of security is maintained as group memberships change.

T3. Token Theft. File server may "steal" the token used by one of its clients and attempt to pass it off to another user. You must develop a mechanism for ensuring that any stolen tokens are usable only on the server at which the theft took place.

Note that you must also address *all* threats from the previous phase of the project. That is, your new functionality should not invalidate previous requirements.

3.2 Mechanism Description

As in Phase 3, the first deliverable for this phase of the project will be a short writeup (3–5 pages should suffice) describing the cryptographic mechanisms and protocols that you will implement to address each of the threats identified in Section 3.1 of this assignment. This writeup should begin with an introductory paragraph or two that broadly surveys the types of cryptographic techniques that your group has decided to use to address threats T1–T3. You should then have one section for each threat, with each section containing the following information:

- Begin by describing the threat treated in this section. This may include describing examples of the threat being exploited by an adversary, a short discussion of why this threat is problematic and needs to be addressed, and/or diagrams showing how the threat might manifest in your group's current (insecure) implementation.

- Next, provide a short description of the mechanism that you chose to implement to protect against this threat. For interactive protocols, it would be helpful to provide a diagram explaining the messages exchanged between participating principals. (See the notes from Lecture 10 for example diagrams.) Be sure to explain any cryptographic choices that your group makes: What types of algorithms, modes of operation, and/or key lengths did you chose? Why? If shared keys are needed, how are they exchanged?
- Lastly, provide a short argument addressing why your proposed mechanism sufficiently addresses this particular threat. This argument should address the correctness of your approach, as well as its overall security. For example, if your mechanism involves a key agreement or key exchange protocol, you should argue that both parties agree on the same key (correctness) and that no other party can figure out the key (security).

After completing one section for each threat, conclude with a paragraph or two discussing the interplay between your proposed mechanisms, and commenting on the design process that your group followed. Did you discuss other ideas that didn't pan out before settling on the above-documented approach? Did you end up designing a really interesting protocol suite that addresses multiple threats at once? Use this space to show off your hard work! As in the last phase of the project, *10% of your grade for this phase of the project is based upon a discussion of your writeup in-person with Professor Lee prior to the project due date.*

3.3 Implementation Requirements

We strongly recommend that you leverage the expertise developed in Homework #1 and use the BouncyCastle cryptography API to incorporate any cryptographic functionality that you may need. Lastly, this project will be graded using the machines in 6110 SENSQ. Please ensure that your code runs correctly on these machines before you submit it.

4 Extra Credit

As in previous phases of the project, you again have the opportunity to earn up to 5% extra credit. Should you happen to complete the required portions of the project early, consider adding in extra functionality in exchange for a few extra points (and a more interesting project). Any extra features that you add will qualify, so brainstorm as a group and see what you come up with! If you opt to do any extra credit, be sure to include a brief description of it when you submit the project (see Section 5).

5 What should I turn in?

To submit your project, first create a tarball containing the following items:

- **Mechanism description.** Include the mechanism description report described in Section 3.2 as the file `mechanisms.[txt|pdf]`. *Only* text or PDF files will be graded!
- **Full source code.** Include *everything* that we need to compile your code from scratch. Please do not include any class files, as we will be rebuilding your code before

we evaluate it. Note that we will be grading your assignment using the machines in SENSQ 6110. It is in your best interest to test your code in that room prior to the submission deadline!

- **Compilation instructions.** In a text file named `compile.txt`, please write detailed instructions for compiling your code. If you opt to use a makefile or other script to build the code, include instructions here on how to run your build script.
- **Usage instructions.** In a text file named `usage.txt`, include explicit instructions on how to use your system. In particular, this should explain how to start your group server and file server, as well as how to start your client application(s). For each operation supported by your client application(s), provide a short statement of how to invoke this operation.
- **Extra credit.** If you have gone above and beyond the requirements of this project, please explain what you have done in a text file called `extra_credit.txt`.

To turn in your project, copy your tarball to the AFS folder `/afs/cs.pitt.edu/public/incoming/CS1653-adamlee/project_4/` before 11:59 PM on Tuesday, April 5th. Late assignments **will not** be accepted! To prevent naming conflicts during submission, please include the netids of your group members in the name of your tarball.

In addition, *each student* should send a brief email to Professor Lee (adamlee@cs.pitt.edu) and the TA (mehmud@cs.pitt.edu) that indicates his or her assessment of each group member's contribution to this phase of the project (e.g., *Bill did 40% of the work, and Mary did 60% of the work*).