CS 1653: Applied Cryptography and Network Security

Fall 2024

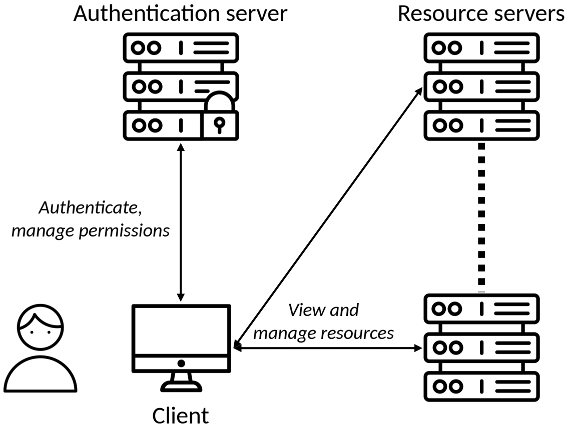
Term Project, Phase 1

Assigned: Tue, Sep 03 **Due:** Mon, Sep 16 11:59 PM

---

1 Background

Over the course of this semester, you will apply the security concepts that are covered in lecture by developing a distributed (networked) system of your own design. Through multiple phases, you will make this system secure against a number of different types of security threats. At a high level, your system must consist of three main components: a single authentication server, a collection of resource servers, and some number of clients.



The *authentication server* will manage the users in the system and keep track of any metadata about each user. This may include groups they belong to, credentials that are applied to them, relationships between them, and any other information that is relevant to determining which resources they will have access to.

Any number of *resource servers* can be deployed throughout the network (without prior approval from the authentication server). The resource servers are responsible for storing and serving whatever objects are managed by your system. They will rely on the authentication server to provide each legitimate user with an authentication and authorization *token* that answers the question, *"Who are you, and what are you permitted to do?"* In this way, the resource servers will not handle (e.g.) passwords or permission changes, and yet they can still be confident that they are serving resources only to those users who are authorized. Multiple resource servers might be used (and trusted!) by different groups of people, may have different purposes or policies, or may communicate with one another so each can retrieve resources stored by the other(s). However, resource servers will not communicate with the authentication server directly.

Users within the system make use of a networked *client application* to log in to the system, manage their user profile (via the authentication server), as well as upload/post, download/view, modify, delete, etc., resources that are stored in the system (via one or more resource servers). Users should have the freedom to choose which authentication server and resource server(s) they want their client application to connect to.

Your project submissions will be made using the git distributed version control system. You are encouraged to use this system to help coordinate your group work, and follow the git best practice: "commit early and often."

2 What do I need to do?

Phase 1 of the course project revolves around forming your group, proposing a system that satisfies the requirements laid out here, and brainstorming how security properties are relevant to your proposed system. The above description is deliberately vague so that you have the intellectual freedom to consider many different possibilities for what resources are managed, how they're viewed, how users are identified, how accesses are granted, and so on. In this phase of the project, you will make decisions around each of these design dimensions.

Task 1: Group Formation

Successfully completing the later phases of this project will require that a considerable amount of time and energy be spent exploring and analyzing system design issues, designing security solutions, and implementing/testing your system. To minimize the burden that this places on each of you and to develop your collaborative development skills, all phases of the project—including this one—will be carried out in groups of 4–6 students.

For your first task, you must form a group with whom to work for the remainder of the semester. Put some thought into your group choice, as your success will depend on one another's commitment to your joint work. Note that all group members will have the opportunity to evaluate the group's performance on each stage of the project.

As you form your group, think about the strengths of your group composition. What are your respective specialities? Where can members fill in for one another's weaknesses? Why do you believe you will work well together? Discuss with each other your planned development process, the roles each member will play, and whether roles will change throughout the semester. Brainstorm how you will manage disputes between group members and your potential strategies for ensuring accountability for everyone's responsibilities. Make sure you're all on the same page regarding expectations of when things will be completed, how often you will meet, and whether you will meet in person or online.

Task 2: System Design

After forming your project group, work with your fellow group members to develop the design of a distributed system that is consistent with the architecture described in Section 1. Consider each of the following as you design your system.

- How will users be identified? How are users created and deleted?
- What metadata is stored about users? How can it be modified?
- What type(s) of resources are stored?
- How are resources accessed or viewed?
- How, when, and by whom can resources be modified?
- What properties of a user determine whether a resource can be accessed?
- How can access be granted and revoked via the authentication server?

In addition to following the system model from Section 1, your system must satisfy the following requirements, designed to ensure you can achieve the learning objectives of later phases.

1. The number of users is not predetermined, and users can be added and removed.
2. Users can complete multiple actions after authenticating, without having to authenticate for each action. Authentication information (e.g., passwords, private keys) is used directly only when communicating with the authentication server to acquire a token; identity and privileges are proved to resource servers via this token.
3. Some aspect of user metadata (groups, roles, attributes, etc.) is used to determine which resources each user has access to. Accesses can be adjusted by modifying the corresponding metadata, allowing multiple accesses to be granted or revoked at once time. (For example, a user can be added to a group to grant them access to all of the group's photos.) This metadata is managed on the authentication server and encoded in the user's token. Resource servers can use this metadata to determine whether accessing a resource should be allowed, but they cannot modify it and should not store it long-term.
4. The authentication server and resource servers can be run on different machines, and can be launched with only SSH access (i.e., via command line).
5. The resources servers do not communicate directly with the authentication server.
6. Any number of resource servers can be launched without approval from the authentication server. Different resources servers may be preferable to different users, so users should get to choose which resource server(s) they want to connect to.
7. Servers must store state, including shared resources, on shutdown.

Finally, note the following implementation constraints, in case they impact your design. We will elaborate on these constraints in the later (development) phases.

You are permitted to develop in any programming language, though the default is Java, and therefore any code examples will be distributed in Java. It is recommended that you develop in a language you are all familiar with, as the time investment to learn a new language *in addition to* all the deliverables for this (very intensive) course may be too much of a commitment. You should also understand that your instructor and TA may not be familiar with your language of choice and therefore may not be able to help with language-specific issues. You should use libraries for cryptographic primitives (e.g., symmetric ciphers, public-key encryption and signing, hash functions) but you **may not** use libraries that implement cryptographic protocols or constructions (e.g., SSL/TLS/HTTPS, SSH, Kerberos, OTR, IPsec, OAuth). You may use libraries for managing your chosen resource type (e.g., images, audio) and/or to serialize information before transmitting it over the network.

Projects will be different in many ways, including in requirements, scope, size, and others. You choose what project to work on, and as such you will take responsibility for it—both the good and the bad. We will not consider the difference between projects as an argument to miss deadlines or negotiate deliverables.

Task 3: Security Properties

Work with your fellow group members to brainstorm a list of *security properties* that are relevant to applications like the one you are proposing. In other words, what might it mean for an application like yours to be *secure*? Remember that security is relative, and that different properties might apply to different use cases and environments. You will likely find properties that are mutually exclusive (e.g., anonymity vs. non-repudiation). You should think about this application from multiple angles, and come up with as many properties as possible that *might* be part of *some valid* definition of security for this type of system.

For each property, come up with (i) a name for the property, (ii) a definition of what this property entails, (iii) a short description of why this property is important, and (iv) any assumptions upon which this property depends. As an example, consider the following property for a group-based file-sharing system:

> **Property 1: Access correctness.** If file f is shared with group g, then user u is able to read, modify, delete, and see the existence of f if and only if user u is a member of group g. Without this requirement, any user could access any file, which is contrary to the notion of group-based file sharing.

The goal of this exercise is to get your group thinking about some the challenges involved with building secure distributed systems. We neither assume that you are experts in applications like the one you're developing, nor that you have prior experience developing secure applications. If you cannot find a place to start, consider what features would make *you* trust the security of such a system, and use this intuition to formulate your initial properties. This will provide you with a starting point that can be refined by examining the features afforded by other systems and reading over relevant portions of your textbooks.

Task 4: Setting Up git and GitHub

You will submit your projects using the git version control system, using repositories distributed via GitHub Classroom. (Note that the project repository will be used for all phases throughout the term!) Although this phase of the project is a writeup and not code, you will use it as an opportunity to familiarize yourself with git. We will not discuss in-depth usage of these tools in class. Thus, as part of this phase of the project, you may need to utilize online resources available for these tools.

Begin this task by familiarizing yourself with the concepts of version control in general, and git in particular. One great resource for this is the Pro Git book by Scott Chacon and Ben Straub (`https://git-scm.com/book/en/v2/`), especially chapters 1.1, 1.3, and 2.1–2.5. Note that you do not need to be an expert in git, but you do need to understand its use cases, including why it is an appropriate tool for this project. Once you are comfortable with the concepts, you should install the git client on your local machine using `https://git-scm.com/` (or your operating system's package manager).

Your git repositories will be stored on GitHub, an online host for git repositories. Each member of your group should sign up for GitHub, preferably using your `pitt.edu` email address for easy discovery. Optionally, you may wish to install a graphical git client, such as SourceTree (`https://www.sourcetreeapp.com/`) or GitHub Desktop (`https://desktop.github.com/`), or set up integration with your IDE.

A template for your writeup will be created for you via a repository on GitHub Classroom.

3 What should I turn in?

The primary deliverable for this phase of your project is a written report that documents your group's activities. This report should have the following structure, as laid out in the provided template:

- **Section 1: Group Information.** List the full name and Pitt email address of each group member. Explain your planned development process, the strengths of your group, the roles each member will play (and whether they will change), and how you plan to enforce internal deadlines and ensure accountability. (See Task 1 for other questions you should consider answering here.)

- **Section 2: Design Proposal.** Overview the design of your system. Define more precisely the roles of the authentication server and resource server, what type of resources are managed, how accesses are determined and modified, what information is stored about users, and the full set of operations that users have access to. (See Task 2 for more details and requirements about your design.)

- **Section 3: Security Properties.** This section should describe the properties that your group has identified as being relevant to the application you are proposing. You should aim to find 15–20 such properties. This section should be arranged as a bulleted list of properties that *may* apply to a system like the one you are designing. As noted above, it is possible that some of these properties will be incompatible with one another.

- **Section 4: References.** If your planned development process, system design, or security properties are inspired by material from outside sources (e.g., books, papers, articles, blog posts, or existing products), your sources should be cited here.

Your reports should be formatted as HTML, following the provided template. This file can be found at `doc/phase1-writeup.htm`. Once you have created your repository, each group member can *clone* it to their own machine to edit the provided HTML file. **Please modify this file only within the designated areas.**

You are encouraged to use this version control repository regularly while collaborating on this project, as each individual's contribution to the group's work will be judged in part by the version control logs. In addition, *each student in your group* will be asked to evaluate the group's performance, including how well everyone is working together and supporting one another.

Your project is due at the precise date and time stated above. We will clone your repository immediately after the due date, so you will be graded on whatever changes have been committed **and pushed** to your repository by this time. Make sure your repository is created and you understanding the submission process well in advance!

After the deadline, we will provide feedback on your writeup, with a special emphasis on the proposed system design. We may suggest that you add or remove features before moving on to Phase 2. Reasons for this include any of the following.

- To scope the work around the components most relevant to our course
- To encourage features that will best enable interesting solutions to the problems that we will pose in later phases
- To avoid scenarios that will make it more difficult for you to achieve the learning objectives of later phases

Please reach out to the course staff if you're unclear on the suggestions we make in our proposal response.