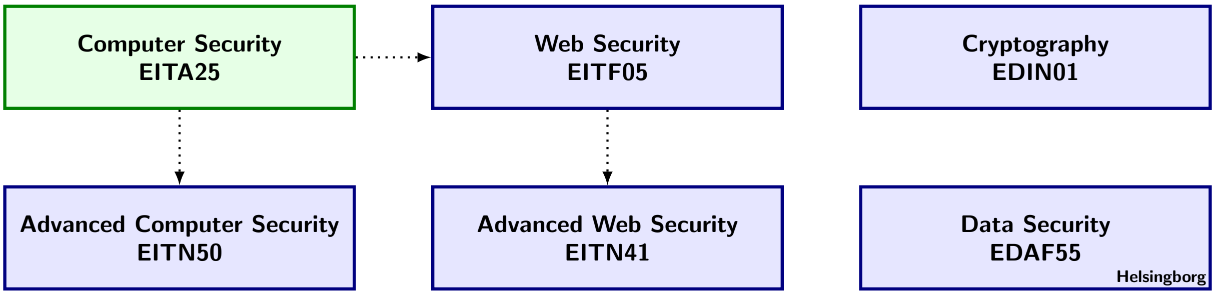
Computer Security 2020

Project 2: Medical Records and Secure Connections

- This project will be done in groups of 3-4 people.

Learning goals:

- Practicing your skills in technical writing and oral presentation.
- Implementing a simple access control scheme.
- Applying knowledge gained in the course by identifying and documenting the strengths and weaknesses of your solution.



1 Introduction

The purpose of this project is to study a “real case” problem that requires security mechanisms in its solution. The project aims at going through most of the stages of the development process, including design, implementation, evaluation and documentation. Medical records contain sensitive data and how these records should be handled is regulated by law [1].

Looking at the law, we can see that

- Someone working at a hospital has access to a medical record only if he/she treats the patient or if he/she needs the medical record for some other reason related to his/her work.
- A hospital can allow the patient to access his/her medical record provided that the patient is authenticated.
- An audit log must be kept that logs all access to a medical record. This can help preventing, detecting and reacting to security breaches and other types of unauthorized access to the medical records.
- Socialstyrelsen can decide that a medical record should be destroyed.

Interpreting the law, “Socialstyrelsen” provides more detailed regulations [2], see also [3]. In these documents, among other things, we can find that

- If an open network is used to manage medical records, the data must be transmitted such that unauthorized disclosure of the data is prevented, and strong authentication must be used, i.e., two-factor authentication.
- People treating a patient must have enough access so that their work can be carried out safely, but they should not have more access than necessary. Only individual access is allowed, no group access.
- The audit log should contain information about who did what and when.

2 Project Description

You will not implement all aspects of all regulations, but the project will use them as a starting point. Let us consider the following scenario:

The medical records in a hospital are kept in a common database managed by a server. Individuals are allowed access to the database by remote access to the server using an open network.

Individuals are of the following types: patient, nurse, doctor, and government agency. Each patient has one or several medical records. Each record contains the name of the associated nurse and doctor (those who treated the patient), the hospital division (where it took place), and some medical data.

For the sake of this project, we interpret the law and the regulations as follows. Each nurse and doctor is associated with a hospital division. Access to medical records is done according to the following rules.

- A patient is allowed to read his/her own list of records.
- A nurse may read and write to all records associated with him/her, and also read all records associated with the same division.
- A doctor may read and write to all records associated with him/her, and also read all records associated with the same division. In addition, the doctor can create new records for a patient provided that the doctor is treating the patient. When creating the record, the doctor also associates a nurse with the record.
- A government agency is allowed to read and delete all types of records.

This interpretation might seem a bit liberal, as it in some cases gives more access than allowed by the regulations. However, deviations of this kind seem to be quite common in real-life systems.

Your implementation should consider a small example with a few patients, a few nurses, a few doctors, and one government agency. Your do not need to include a database system for your records (if you don't want to), simply store them in some convenient way. You must also make sure that all actions are properly logged.

Access to the records is provided through a public client program (you must consider all implementations as public knowledge in your security evaluation). The server is situated in a physically protected room. Access to this room is only allowed under the supervision of trusted staff. Proper backup is assumed.

Think about your model for access control. Some have been covered in the lectures. Which is most appropriate?

Where does the reference monitor concept fit into your access control model?

The project must be implemented in the Java language. You will need to write a client application, a server application and possibly some additional application. You may reuse the client and server implementations from Project 1. Some form of two-factor authentication of individuals is needed and one way to accomplish this is to authenticate users with certificates. This will require them to have both the actual keystore and also the password to the keystore. You are free to use some other two-factor authentication method if you want. All certificates must be signed by a CA, as done in Project 1. Define and use a sensible naming convention for your certificate fields. The communication between the client and the server must be encrypted and established using the network standard TLS. The CA certificate should be installed in a truststore on both the server and the client. The CA private key should be used to sign all other certificates. The server should store all records and respond to client requests. The records do not need to be stored encrypted.

2.1 Peer-review of the functionality

You will perform one functionality peer-review and two peer-reviews of written reports. The functionality review is described here, while the report review process is described in Section 3.4.

Project groups are divided into clusters of 4; groups 1-4, 5-8, and so on. If not divisible by 4, some cluster(s) may contain 1 group more or less. Project groups and cluster groups are listed on the course home page.

Functionality reviews are to be performed pairwise within the cluster. You may subdivide your cluster in any suitable fashion. If you cannot all agree on a group pairing, the default assignment is $(i) \leftrightarrow (i+1), (i+2) \leftrightarrow (i+3)$. That is, groups i and $i+1$ review each other, and so on. If the number of groups in your cluster is odd, then the three last groups form a review cycle according to $(i) \rightarrow (i+1) \rightarrow (i+2) \rightarrow (i)$.

All groups members must actively take part in the functionality review. The expected output from a successful functionality review is a signed functionality review form. This form is available on the project home page (see projects), and it includes instructions on how to perform the review itself.

3 Documentation

The expected documentation output is a report bundle in the form of a pdf-file, about 11-12 A4 pages long, as specified below with approximate page counts.

• Front page with the names of the group members. (1 page)	report	report bundle
• High-level architectural overview. (2 pages)		
• Ethical discussion. (1 page)		
• Security evaluation of the design. (2-3 pages)		
• Peer-reviews of your report. (2 pages)		
• Improvement sheet. (1 page)		
• Signed functionality review form. (1 page)		
• Signed contribution statement form. (1 page)		

You report and all other written documentation must be written in **English**. You are encouraged to use a TeX editor (TeXstudio[1], Overleaf[2],. . .) to write the report and compile the report bundle above into a single pdf-file. This is particularly encouraged if one or more of your group members have never used TeX before.

Please use single spacing and a normal-sized font with serifs in your documentation! Playing around with large fonts or spacing just makes your report look weak, do not fall into that trap. Make sure that your report includes references to all books, articles, web pages, pictures and such that you have used for making your program or fuelling your arguments and discussions. If you have not included a reference, then you are claiming that you are the original author. And do not use copyrighted material (images, . . .) without permission!

3.1 High-level architectural overview

The first part is a short *high-level* architectural overview that explains how your program is *structured.* The level of your description is the important focus here. Motivate your design choices. Do not go into insignificant details. Upon reading your architectural overview, the reader should clearly, correctly and quickly understand the structure of your system.

You are required to draw an image to illustrate this structure (half page). Be sure to illustrate how keystores and truststores are used, by whom and how, certificate hierarchy, and so on. Define the main components and actors, and show how information flows. Explain your access control scheme.

Your main challenge here is to use your technical writing ability to explain technical details in a way that is easily absorbed and not so easily misunderstood. Keep your description at a high structural level. Only include details if they are important from a security perspective. Do *not* include a user guide or user instructions for your program!

The intended audience for this part should be a student in your program that has not taken the Computer Security course. Upon reading your description, the reader should understand the structure and main points of your program.

3.2 Ethical discussion

Your implementation focuses on confidentiality. This is the implementation that the fictional hospital administration has commissioned from you. In reality, availability is much more important than confidentiality (arguably, but do you see why?). Do your responsibilities as an engineer and security expert stretch beyond delivering the requested product?

How would you implement access control that is suitable for live production in a real hospital environment? Formulate an access control scheme (define it, do not implement it) that provides the best

[1] `https://www.texstudio.org`

[2] `https://www.overleaf.com`

tradeoff between confidentiality, integrity and availability. Compare this access control scheme to the one you have implemented here and list advantages and drawbacks of each scheme.

Discuss the ethical considerations that need to be weighed when designing and using a system in the health care industry. For the three players

- hospital administration (ordering the system),
- engineer / security expert (you, designer/implementer),
- hospital staff (system end-users),

describe which aspects (such as economical, patient health, availability, privacy, system vulnerabilities, an so on) that could or should be ethically weighed against one another. For example, budget goals and patient safety can easily be seen to be opposing factors in some cases.

You may consider the articles [5, 6].

3.3 Security evaluation

The list of security consideration should list all attack types and major security issues you can think of and explain if, why and/or how it is or is not applicable to your system. For each attack/issue, explain clearly and concisely what you have done to protect your solution. Or briefly explain why no protection is needed. Your report should clearly motivate your security related design choices. For this part, explaining how and why you have implemented two-factor authentication the way you have is a requirement. In the same way as for the architectural overview, aim for clarity, correctness and assimilation speed. That is, a reader should quickly be able to grasp the security status of your system.

In this part of your report, include a little something on cipher suite selection. Which cipher suite is chosen when you run your program? Dissect the cipher suite identifier and explain its different parts. Are there good and bad cipher suites? How can you control the choice of cipher suites in your programs?

3.4 Double peer-review of the report

You are to produce peer-reviews of two written reports, and your report will be reviewed by two other groups within your cluster. The process is as follows.

When group i has written their project report, they distribute it *to* groups $i + 1$ and $i + 2$ according to $(i) \rightarrow (i + 1, i + 2)$. Consequently, group i will receive and review written reports *from* groups $i - 1$ and $i - 2$, following $(i) \leftarrow (i - 1, i - 2)$. And of course, group numbers wrap around within the cluster.

A review is to be written densely on at most one A4 page. Verify that the report achieves the goals as specified in this instruction. Give encouraging feedback on the good parts, and identify the major points of improvement. You may check for structure, language, technical correctness, readability, and so on. Be constructive. Suggest ways of improving the report. You do not need to provide a general grade or rating.

When you have received the reviews of your report, read them and update your report accordingly. List or summarize the actions you have taken to improve your paper on at most one A4 page (improvement sheet).

3.5 Contribution statement

Individual project contributions of each group member are to be listed on the Contribution Statement Form, which is available on the course home page. All group members must sign the contribution statement.

Actively discussing and learning is also a way of contributing. Allow for the varying backgrounds of your group members. However, repeated failure to show up for meetings, or ”failing” communication is not acceptable.

3.6 Submitting

Last but not least, bundle up your report together with the reviews of it, the improvement sheet, the signed functionality review form and the contribution statement form. Hand it in electronically as *one* uncompressed pdf-file, email it to Syafiq (syafiq_al.atiiq@eit.lth.se). Write 'EITA25 P2' (without quotes) in the subject line.

4 Presentation

Your group is to perform a short but coordinated oral presentation in a mini-conference format. The language of the presentation is **English**. The main focus for this part is your presentation technique. How well can you handle giving a technical presentation within a small given time frame?

Other groups from your course, and Syafiq, will be your audience. Decide on a presentation time that works for all group members. Match your schedules and book a time slot with Syafiq. Some available time slots will be shown on the course home page.

Also rank the topics that are labelled A to G on the course home page in your order of preference. Your group number and a topic letter will be listed on the course home page when you have been assigned a time slot. The topic letter determines the focus of your special topic presentation. Your aim for this part is to present the topic and to teach your audience something *new, advanced or exciting* – something that they **did not know before**. Dazzle and intrigue your audience! If possible, relate the topic to your system.

The presentation itself is to be 6 minutes long (you will be timed) and contain/show the following.

- A brief architectural overview of your implementation. (optional)
- *Convincingly* show that you have successfully set up a TLS connection.
- *Convincingly* show that your system is working and that it fulfills the given requirements.
- You are required to include actual code in your presentation.
- Topic focus as described above.

All groups members must contribute to the presentation, not necessarily by speaking. For example, one resource may be dedicated to running a live demo and/or the slide show while the others present. Attending your own session is mandatory. You may attend other sessions for inspiration, but this is optional.

Your main challenge here is the given time frame, so make your presentation concise and coordinated. Aim at making the presentation seem relaxed. You will need to rehearse your presentation several times to achieve this goal.

If you think that you can pull off a real-time demonstration of your code, please do so. This often makes a presentation more interesting. But make your presentation robust, so that you have presentation slides as a backup if something unexpected happens. A fully slide-based presentation is also an acceptable option. In either case, you need to convince your audience that you have a fully functional implementation (and more so if you do not sport a live demo).

After your presentation, Syafiq and the other groups will give feedback on your presentation. Did you achieve the presentation goals? Comment on efficiency, accessibility, flow, credibility, presentation techniques, future improvements, and so on. This procedure will be detailed when the session starts.

Practice setting up your presentation quickly. The time schedule is as follows.

0. Setting up first presentation. (3 minutes before scheduled session start)
1. Introduction by Syafiq. (3 minutes)
2. Presentation + feedback. (6+5 minutes)
3. Setting up next presentation. (2 minute)

Steps 2 and 3 are repeated as necessary.

Email the presentation file (pdf, uncompressed) to syafiq_al.atiiq@eit.lth.se the day before your presentation, deadline at 23.59. Write 'EITA25 P2' (without quotes) in the subject line. Syafiq will bring your pdf on his computer, and make it available for presentation. In other words, you do not need to bring your own computer if your presentation is purely pdf-based.

If you still need to borrow a computer for the presentation, synchronize with one of the other cluster groups, or let Syafiq know well in advance.

5 Delivery summary and deadlines

Your group will need to provide or do the following:

1. Book a presentation time for your group (rank time slots and topic letters). (group manager)
2. Distribute your written report to your reviewers, cc to Syafiq. (group manager)
3. Return report review comments, cc to Syafiq. (group manager)
4. Functionality review completed.
5. Mail code, configuration files (compression ok) and report bundle (pdf, uncompressed) to Syafiq. (group manager)
6. Mail presentation to Syafiq. (group manager)

Deadlines as listed on the course home page.

6 Resources

- If you do not have it already, the Java SDK package can be downloaded at [7].
- Java security documentation can be found at [8]. The most important part for you is the JSSE documentation which can be found at [9]. This is a rather large document but you will learn tons of stuff by reading it and it makes the project much easier. Spending time reading the JSSE documentation will probably save you lots of time in the end. The corresponding document is more extensive for JSE 9, so we point you to 8. You can use anyone of them.
- There is sample code in the JSSE documentation. You should be most interested in `ClassServer.java`, `ClassFileServer.java` and `SSLSocketClientWithClientAuth`.
- Java SE 8 API specification can be found at [10]. (You can replace 8 by 9 in the link to get the specification for JSE 9.) Java SE 11 API specification can be found at [11].
- The java language specification can be found at [12].

7 Hints and other useful information

The focus of this course is on computer security, and we use programming as a tool to explore security topics. We do not require you to write a program that is spotless in all regards, but we do require that you think extensively about all security aspects that you can find, both technical and nontechnical. In your program, functionality is more important than design. We will not be selling your solution to investors, so you do not need to spend your valuable time making your program look nice. In fact, a GUI is *entirely* optional.

Here are some useful hints.

Hint 1: Take advantage of what you learned in Project 1 when creating keystores and truststores for different end-users. The keytool parameter `-dname` can be useful to minimize tedious and error-prone human interaction when generating certificates.

Hint 2: Before you begin coding, draw a design sketch to show which keystores and truststores your design will be using and where. This sketch can also be used for your report.
Hint 3: When you have established the connection, the server might want to read the client certificate to check which user it is. Check your source code to see how this was done in Project 1.
Hint 4: Distribute the work within the group. Do not sit 4 people in front of the computer.
Hint 5: The JSSE documentation can be tough to read, but it is a *very* good source of information with explanations and examples. Consulting it may save you lots of time.
Hint 6: When starting a Java program, a truststore can be specified by setting the system properties javax.net.ssl.trustStore and javax.net.ssl.trustStorePassword as

```
java -Djavax.net.ssl.trustStore=theTruststore -Djavax.net.ssl.trustStorePassword=passphrase
prog
```

but it can also be set in the source code. If you reuse Project 1, check your source code to see how it was done there.
Hint 7: We recommend using java version 8 and above.
Hint 8: How do you efficiently deliver a speech or presentation that makes your audience see you as knowledgeable and well-prepared with a professional attitude? It is not necessarily difficult. Consider the effects of the following advice. Make sure that you know what is on *the next* slide in your presentation – before it appears. You save time because you know what to say ahead of time, so your presentation seems efficient and well-coordinated. You will also seem professionally well-prepared, because you do not have to look at your own slides to remember what to say.

References

[1] Patientdatalag (2008:355), `https://lagen.nu/2008:355`, last accessed on 2020-02-03.

[2] HSLF-FS 2016:40, Journalföring och behandling av personuppgifter i hälso- och sjukvården, `https://www.socialstyrelsen.se/Lists/Artikelkatalog/Attachments/20494/2017-3-2.pdf`, last accessed on 2020-02-03.

[3] Gemensamma författningssamlingen avseende hälso- och sjukvård, socialtjänst, läkemedel, folkhälsa m.m., `https://www.socialstyrelsen.se/Lists/Artikelkatalog/Attachments/20165/2016-4-44.pdf`, last accessed on 2020-02-03.

[4] NIST. Guide to Attribute Based Access Control (ABAC) Definition and Considerations. `http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-162.pdf` Last accessed 2019-02-11.

[5] Datainspektionen. Bristande åtkomstkontroll på Karolinska sjukhuset, 2009-04-03,
`http://www.datainspektionen.se/press/nyheter/2009/bristande-atkomstkontroll-pa-karolinska-sjukhu`
last accessed on 2019-02-11.

[6] Sydsvenskan. Undersköterska kollade i journaler, 2016-02-02,
`http://www.sydsvenskan.se/lund/underskoterska-tjuvkollade-i-journaler/`, last accessed on 2019-02-11.

[7] Oracle, Java SDK, `http://www.oracle.com/technetwork/java/javase/downloads/index.html`, last accessed on 2019-02-11.

[8] Oracle, Java SE documentation, `http://download.oracle.com/javase/8/docs/technotes/guides/security/index.html`, last accessed on 2019-02-11.

[9] Oracle, JSSE documentation, `http://download.oracle.com/javase/8/docs/technotes/guides/security/jsse/JSSERefGuide.html`, last accessed on 2019-02-11.

[10] Oracle, Java SE 8 API documentation, `http://download.oracle.com/javase/8/docs/api/`, last accessed on 2019-02-11.

[11] Oracle, Java SE 11 API documentation, `https://docs.oracle.com/en/java/javase/11/docs/api/index.html`, last accessed on 2019-02-11.

[12] Sun, Java Language and Virtual Machine Specifications, `http://java.sun.com/docs/books/jls/index.html`, last accessed on 2019-02-11.