THE UNIVERSITY OF WESTERN AUSTRALIA

School of Computer Science and Software Engineering

CITS3002 Networks and Security

Practical project 2016

(due 11:59pm Sun 29th May - end of week 12)

See also: **Clarifications** and **Resources**.

Digital signatures are increasingly used to verify the authenticity of electronic documents. Digital signatures are constructed by calculating a cryptographic hash of a document, and encrypting that hash with the signer's private key. The recipient of the document may consider it to be authentic or 'safe' (depending on the context) if they can verify the digital signature.

Digital certificates are then used to verify the identity of the individual presenting the public key with which a digital signature may be verified. Digital certificates are, themselves, verified by following a hierarchical **chain of trust** from the original certificate to a root authority. The root authority "earns" their trust through both being an early member of the "certification marketplace", and through their continued survival in that competitive marketplace.

An alternative to the hierarchical certificate model is provided by a **circle of trust** in which we believe that a signed document is authentic or 'safe' if it has been signed by someone whose digital certificate has been signed by someone, whose digital certificate has been signed by someone, ..., until we locate a closed circle of people who each vouch for each other. No top-level definitive root authority is required. Instead, trust is earned if the closed circle is simply long enough.

Aim

The aim of the project is to develop a network-accessible secured file facility, named *oldtrusty*, whose contents are protected by a "circle of trust". The files being managed may be stored on any conventional file-system (of your choosing). All activities involving the files, such as adding, fetching, and verifying, are to be requested by a command-line client, supporting the following command-line options:

-a filename	add or replace a file on the *oldtrusty* server
-c number	provide the required circumference (length) of a circle of trust
-f filename	fetch an existing file from the *oldtrusty* server (simply sent to *stdout*)
-h hostname:port	provide the remote address hosting the *oldtrusty* server
-l	list all stored files and how they are protected
-n name	require a circle of trust to involve the named person (i.e. their certificate)
-u certificate	upload a certificate to the *oldtrusty* server
-v filename certificate	vouch for the authenticity of an *existing file* in the *oldtrusty* server using the indicated certificate

All communication between the *oldtrusty* client application and *oldtrusty* server must be secured by secure socket layer (SSL) channels. You are encouraged to use the formal X509 digital certificate format, for which a number of libraries and tools exist, although you may choose to develop your own (simpler) format.

A significant part of this project requires you to determine under what circumstances, and in what combinations, the above command-line options are reasonable and necessary. For example, you'll need to identify and answer questions such as "how and where are certificates managed?", "how and where are the files signed?", and "how is a circle of trust maintained, determined, and reported?".

There is *a lot* of relevant well-written code and tutorial documents available on the web. Pointers to these will be added a **project resources** webpage. You will need to devote time to reading and understanding this material, but you will not need to develop significant amounts of new code.

Important dates and project submission

The project contributes **40%** of your mark in CITS3002 this year, and has a number of important dates and deadlines:

- by **5pm Thursday 26th May** (week 12)
  complete the project demonstration booking sheet, which will be pinned up outside of CSSE Lab 2.3.
- by **11:59pm Sun 29th May** (end of week 12)

submit via cssubmit your team's project submission. By this deadline submit all source code, digital certificates, and scripts that you wish to be assessed.

- during **the week commencing Monday 30th May** (week 13)
  project demonstrations.

Constraints

The **constraints** of the project require that:

1. you may develop your project in Java, C, or Python, supported by their functions (methods, classes, libraries, ...) for the OpenSSL suite or Java's SE Security Platform.
2. you will need to write at least two distinct pieces of software, a client and a server.
3. your client and server programs must be written in different programming languages.
4. your project must support two or more 'simultaneous' connections between the server and clients, though the server need not service multiple clients concurrently.
5. your project must execute on/across at least three different networked computers (not simply on a single computer, although it may be developed on just one).
6. your project may be developed for Linux, Windows, or Mac-OSX (or a combination).
7. all network traffic must be encrypted using SSL.
8. your client program does not need to provide a graphical interface (GUI).

Project demonstration

Your team must also arrange a demonstration of your software, for **up to 30 minutes, in week 13 (or possibly week 14)**. During the demonstration, you'll be logging into your account on your chosen operating systems. A booking sheet will be provided closer to the deadline. During the demonstration, your team should:

- briefly describe design decisions and assumptions that you have made in your project. You must clearly identify what is being protected by your system, how that protection is assured, and identify any currently known weaknesses with your approach or its implementation.
- re-compile your programs, and initialize and invoke any server and/or client programs. Describe the contents of each necessary directory and its files.
- demonstrate, through a small number of examples, how someone may use your software system.
- do not prepare a PowerPoint-style presentation.

Working in teams of up to 3

The project may to be undertaken in teams of up to 3 students (no, not 4). The motivation for working in small teams is to enhance communication skills amongst students, and to enable you to attempt a project considered of greater difficulty than would normally be reasonable for the time available. It is anticipated that this project will require **20-35 hours** of study by each member of the 3-person teams.

The project is worth **40%** of your mark in CITS3002 this year, and the distribution of marks within your team (typically one-third each) must be agreed to by all members of your team.

Only one team member needs submit files using cssubmit. Ensure that all students' names and student numbers appear in all submitted materials.

Anyone needing to find a project partner should read *Students seeking project partners* as soon as possible.

Clarifications

Please post requests for clarification about any aspect of the project to *help3002* so that all students may remain equally informed. Clarifications will be also added to the **project clarifications** webpage, and additional materials may be addded to the **project resources** webpage.

Good luck,