CMPE 202 Class Project, Fall 2018

The document may change at times to add more details or to correct any mistakes. An announcement will be made if the document is updated.

Table of Contents

All students should work in a group. A group should have between 3-3 members.

Although inspired by an existing product, it is not a real application. This application is not intended to be fully functional, rather it is to be used as a backdrop for the secure software development activities that are planned in the class project.

I have intentionally not included any architecture and data flow diagrams in this document. Doing it would have taken away the threat modeling activity part of the project from you.

The project consists of:

- This document
- A data file (SCFFile.tlv)
- A Windows executable (SCFParser.exe)
- Two Windows DLL files (libeay32.dll and ssleay32.dll) - They are needed to execute SCFParser.exe

The project deliverables are:

1. A report
2. Group presentation
3. Demo your program and fuzz testing activity

The report must contain the following:

1) Answers to the questions in the Questions section of this document.
2) Output of the static analysis, before and after the fixes are applied.
3) Output of the fuzzing analysis.
4) Include application code in Appendix section of the report.

Project Definition

1. Overview

Softcorp inc. offers a monitoring system that consists of **sensors** that collect environmental data and **appliances** that process them. The sensors communicate with 2 services running on an appliance: a **Web Server** and a **Controller**. Many appliances can be set up for redundancy purposes.

The communication between the sensors and the services is TLS protected. To keep the solution simple, a PKI is not required. Instead, the services certificates are bundled in SCF (Signed Certificates File) and the sensors download it from a **TFTP** server. The SCF contains the services certificates. The sensors only trust the certificates contained in the SCF.

The following sections detail the different components of the solution.

2. The Sensors

The sensors are small devices that collect environmental data like the temperature and the humidity level. They are configured out of the box via a console interface. The console allows the configurations of the console access password, the device name and a **unique secret string** that the device uses to authenticate to the 2 services running on the appliance(s).

Network configuration settings (IP address, DNS, TFTP server, gateway) are obtained via DHCP.

Other configuration settings (e.g. frequency of the data collection, the appliance hostname) are contained in a signed configuration file that is downloaded from a TFTP server.

The **SCF** is also downloaded from the TFTP server and is used to validate the appliance services certificates.

The sensors have a small display screen. The sensor sends an https request to the **Web Server** with its name, secret string and the collected data and gets an html page back that gets rendered on the screen. The html page reflects back the environmental data and the name and may contain other useful information that depends on the geographic location of the sensor.

The sensors communicate also with the **Controller(s)**. The communication uses a custom bi-directional protocol on top of a permanent TLS/TCP connection. The first message that sensors send after a connection is established is a registration message containing a sensor's name and its secret string. After the registration is accepted, the sensor starts

sending status information (e.g. battery level, sensor's hardware health). The controller uses the connection to instruct sensors to perform some actions (e.g. change the frequency of data collection, sound off an audio signal, etc.).

3. The Web Server

The Web Server runs on the appliance and provides 3 servlets:

1. A **collection servlet** that the sensors send data to.

2. A **configuration servlet** that an admin user uses to configure the appliance and the sensors (names, secret string, settings) and check the health of the appliance and the sensors. It's also used to customize the page that gets rendered on the sensors screens. The configuration information is stored in an Oracle database.

3. A **reporting servlet** that provides a UI for users to view the collected data in a variety of ways.

The collection servlet gets the collected data from the sensors and stores them in an **Oracle database**.

4. The controller

The controller is a C++ process that listens on TCP port 3333 and has permanent TCP connections with the sensors. The Controller gathers status information from the sensors and stores it the Oracle database.

The controller also checks the Oracle database for any sensor settings changes and communicates them to the sensor through the connection.

5. The Sensor Configuration File

The configuration file is stored on a TFTP server and contains the default configuration settings for the sensors. The configuration file is signed with the private key of one of the Controllers. The sensors validate the signature using the Controller's certificate contained in the **SCF**.

The format of the configuration file and how it's built and moved to the TFTP server are not in scope for this project.

6. The SCF (Signed Certificates File)

The SCF contains the certificates of the Web Service and the Controller service. Note that if redundancy is supported, the SCF would contain more than 1 certificate for each service. The SCF is signed with one of the controllers' private key. The SCF is stored on a TFTP server. The building of the SCF and how it is moved to the TFTP server are not in scope for this project.

The sensors always validate the signature of the SCF by making sure that the signature can be verified with one of the Controllers' certificate. They trust the certificates in the very first SCF that they download in a leap of faith. From then on, a key whose certificate is in the existing SCF must sign any new SCF.

The SCF is composed of a header followed by a body. It uses a TLV (Type, Length, Value) format.

The SCF Header

The SCF header contains envelope information about the file as well as the signature. The format of the header is presented in figure 1.

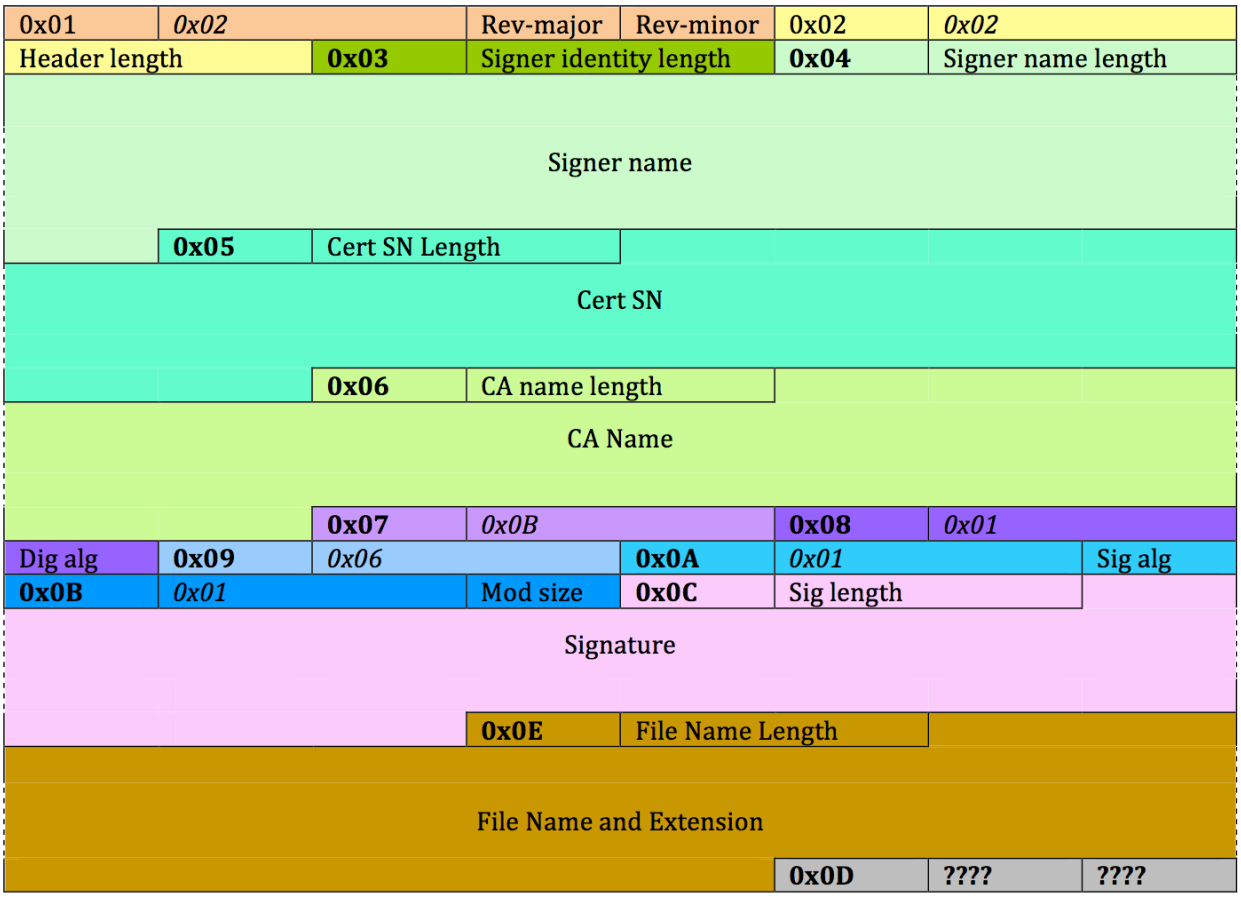

Figure 1: The SCF Header

The header is a suite of fields. Each field contains the field type (e.g. 0x05), length (L) and value (V).

The type is always one byte and the length is two bytes.
Note that the order of the different fields is not important (except for the opening tags and the revisions) and that not all fields are required.

The SCF Body

The SCF Body contains a list of **records** for each certificate. A record consists of the certificate, the certificate's role (Controller or Web Service) and other information. The format of each record is presented in figure 2.

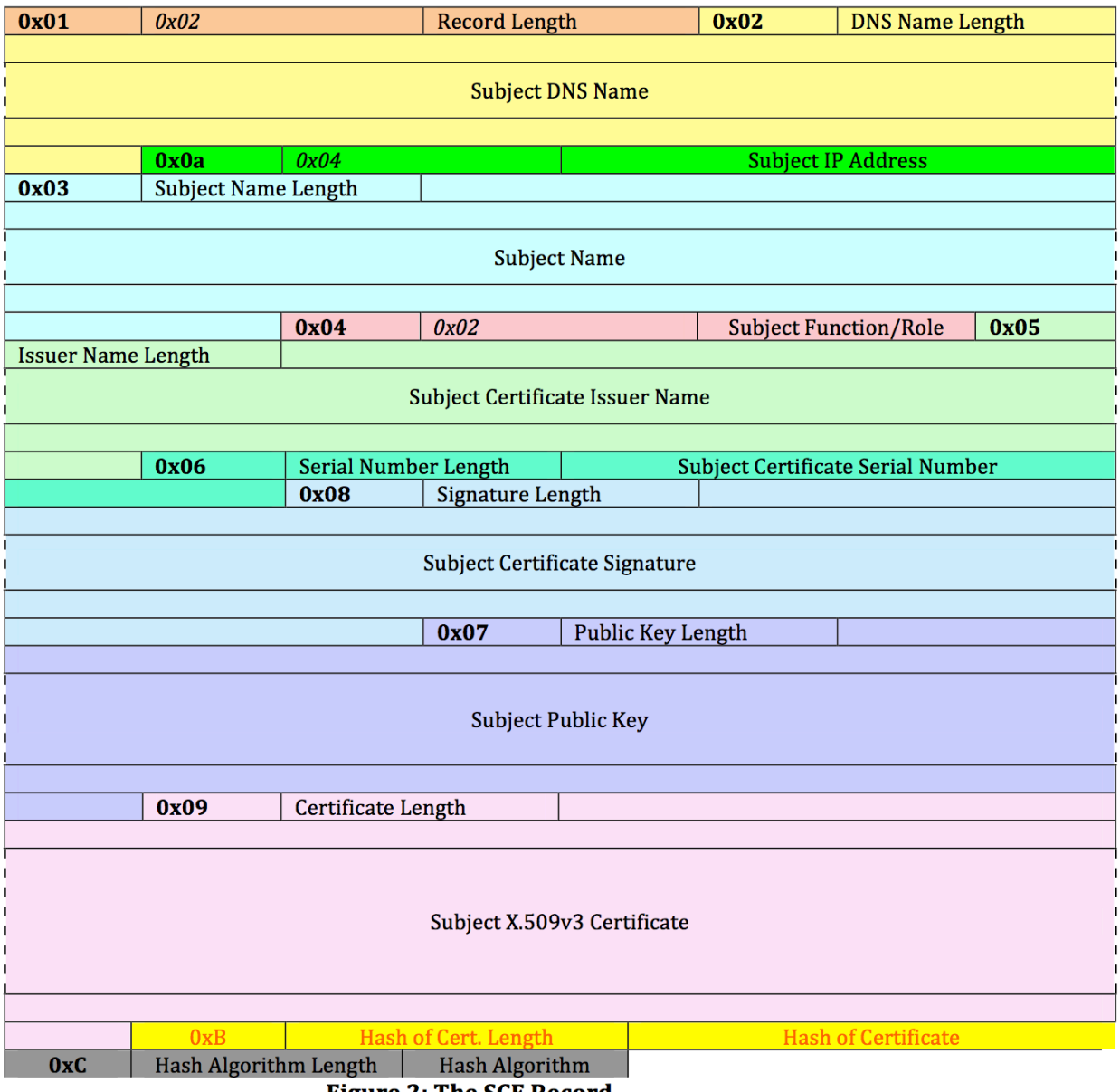

Figure 2: The SCF Record

Each record contains a list of fields. Each field contains the field type (e.g. 0x07), length (L) and value (V).

The type is always one byte and the length is two bytes.
Note that the order of the different fields is not important and that not all fields are required.

The required fields are the record length, the Subject/Function role and the x509 certificate.

7. Third Party Software

The application uses the following 3rd party software:

- Openssl 1.1.0
- Apache Tomcat 8.0.3
- Oracle 12.1
- BouncyCastle 1.52

SDL Activities

Each group needs to perform the following activities:

1. Security Baseline

Create a list of security baseline requirements for the project. Make sure to include all components.

2. Threat Modeling

Perform threat modeling on your project.

Model all the interactions and external dependencies you that can think of. However, you do not need to model “infrastructure” flows such as DNS or DHCP queries.

3. Source Code Static Analysis

For this task, you will write a small parsing program (a smaller version of SCFParser.exe).

You can write it in any language you choose. You will need to run static analysis on the program.

Parsing Program

Your program needs to parse an SCF file (use SCFFile.tlv for your testing) and extract all the certificates with their functions. Additionally, it needs to extract the CN of the certificates.

Static Analysis

Identify an open source static analysis tool for your language of choice and run it on your program. Include in your report the outputs of the tool when you run it the first time, and when you run it again after you have fixed the security defects found in your first run.

4. Fuzzing

In this activity, you will use the SCFFile.tlv SCF file and the SCFParser.exe executable. Put the two Windows DLL files in the same directory as SCFParser.exe and SCFFile.tlv file.

The SCFParser parses any SCF file and display the contents in an easy to read dump. The SCFParser simulates the parsing that the sensors do to extract information from the SCF.

Try it in a Windows terminal:

C:\CSFParser SCFFile.tlv

and observe the output.

Your task is to try to crash the parser by fuzzing the file passed as a parameter to it. If you manage to crash the parser, great☺ However, the focus is on the methodology that you use to fuzz the input.

Your report must explain the steps you took and the strategy you used to fuzz the input. Your strategy can either leverage a tool or be completely manual. If you manage to crash the parser at your first attempt, please still document what else you could have done.

5.Third-Party software tracking

List all third-party software/libraries that you use in your program. Ensure that you track them in order to determine new vulnerabilities discovered in them.

Identify any CVEs associated with these software/libraries.

Questions

1. Custom Sensor/Controller Protocol

You are tasked with designing the custom sensor/controller protocol. Explain in at least a half page the steps you would take to design a secure protocol.

2. SDL Activities

Can you think of any other SDL activities that would be needed for this project beside the ones covered in this project? Explain why you think they are needed.

3. SDL Metrics

Meaningful security metrics allow organization to determine the effectiveness of its security controls. List the SDL metrics that you would capture in this project and explain how they would help bring maturity to your SDL process.

4. Changes to the Solution

A. Softcorp decides to add more fields in the UI page of the configuration servlet. Which SDL activities would this trigger?

B. Softcorp now plans to add a C++ process to its appliance. The process would use the sensors data in the Oracle database and send SNMP traps if some concerning trend is identified in the data set. Which SDL activities would be triggered in this case?

Project Grading Rubric

Task # & Description	Points (100)
1 - Security baseline	10
2 - Threat modeling	15
3 - Static analysis	20

Task # & Description	Points (100)
4 - Fuzzing	10
5 - Third-party software	5
Answers to questions	20
Demo (Fuzzing + program)	10
Presentation	10

Appendix A