BSP Project Description:
< PROJECT_TITLE >

Tuesday 24th October, 2023 - 12:49

Meireles Lopes Steve
University of Luxembourg
Email: steve.meireles.001@student.uni.lu

This report has been produced under the supervision of:
Skrobot Marjan
University of Luxembourg
Email: marjan.skrobot@uni.lu

Abstract

This document is a template for the project subject description (PSD for short) report that is to be delivered by any BiCS student at the end of the phase 1 of a Bachelor Semester Project (BSP). The Latex source files are available at:
https://github.com/nicolasguelfi/lu.uni.course.bics.global

This template is to be used using the Latex document preparation system or using any document preparation system. The whole document should be 2000 words (± 20% with progressive penalties) for S2 to S6 students and 3000 words (± 20% with progressive penalties) for S1 students (excluding the references and annexes) and the proportions must be preserved.
Remove/replace all generic and template specific texts when you create your document from this template!

A tutor (or any person having contributed to the BSP work) is not a co-author per se for a student's work. It is possible to exploit a BSP report to produce a scientific and technical publication. In this case, the authors list has to be discussed and agreed with the concerned parties.

1. Main required competencies ([5%..10%] of total words)

1.1. Scientific main required competencies

For the scientific deliverable the needs to know the basics of cryptography and security such as password security, symmetric and assymetric encryption and key exchange. In addition to that he requires math knowledge such as set theory and math notations.

1.2. Technical main required competencies

The technical deliverable requires the student to know the same topics as for the scientifc deliverable. Moreover he will have to be able to program in the programming language python.

2. A Scientific Deliverable 1 ([±40%] of total words)

A lot of enterprises use username and password logins to authenticate their users or employees. Everybody has a few technologies that require them to authenticate with their username and password. It is commonly known that this is often exploited by attackers through weak passwords, brute forcing or even pishing attacks. This brings us to the scientific question: In the context of client-to-server authentication over an insecure network, how does one detect if a password file on the server side (containing usernames and passwords) was compromised by intruders and at the same time prevent phishing attacks on clients?

To answer this question the deliverable will study password security, secure storage of password and brute forcing attacks. It will also study the honeyword paper from Riverst Juels to find technologies to enhance the security propereties. Furthermore, it looks into ways to enhance current password authentication protocols from sending a hash of a user's password over TLS to utilzing PAKE which prevents pishing attack. On top of all that the paper is going to examine the SweetPAKE protocol from Arriage, Ryan and Skrobot which uses the PAKE and enhances it with decoy password which adds password file leakage deetection mechanism at the server side.

3. A Technical Deliverable 1([±40%] of total words)

The description of what will be presented in the technical deliverables section of the final report. This section must present and be based on a state of the art of the topics addressed by the technical aspects of the project.
DO NOT PRESENT THE IMPLEMENTATION OF THE PRODUCT IN THIS SECTION

After having given a 1 to 3 lines for the name and main goal of the product, it is advised to structure this section in sub-sections and sub-sub sections. A possible structure could be:

1) Context: the technical context of the product: here it should be presented the links bewteen the targeted product and the existing comparable tools and related technologies, and provided motivations for this product.
2) Functionalities: the sub-components/sub-products deduced from the list of functionalities clearly described per component for the product targeted. This section corresponds to the ”Requirements” of the product.
3) Ideas: the ideas/directions/type of solution you would propose to produce the product. In this subsection, it must be presented an overview/high level understanding of what is expected to be the product produced. Some wireframe / prototype of the GUI can be used to the solutions for the product services/functionalities. Libraries used and general code overviw can be explained. This section corresponds to the ”design” of the product.

In the three sub-sections proposed it must be provided a commented use of the relevant references on which the work is proposed to be based on.

References

[BiCS(2021)] BiCS Bachelor Semester Project Report Template. https://github.com/nicolasguelfi/lu.uni.course.bics.global University of Luxembourg, BiCS - Bachelor in Computer Science (2021).

[BiCS(2021)] Bachelor in Computer Science: BiCS Semester Projects Reference Document. Technical report, University of Luxembourg (2021)

[Armstrong and Green(2017)] J Scott Armstrong and Kesten C Green. Guidelines for science: Evidence and checklists. Scholarly Commons, pages 1–24, 2017. https://repository.upenn.edu/marketing_papers/181/

4. Appendix

All images and additional material go there depending on your specific available data.

4.1. Source Code

The following environment shows the correct and mandatory way to insert your code.

Listing 1: Caption example.

```
import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None #to become the incidence matrix
    VT = np.zeros((n*m,1), int) #dummy variable

    #compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m-1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

                if M is None:
                    M = np.copy(VT)
                else:
                    M = np.concatenate((M, VT), 1)

                VT = np.zeros((n*m,1), int)

    return M
```