CIS-481: Introduction to Information Security

InfoSec Chapter Exercise #7

Team: Project Team 11
Participants: Sohal Patel, Erika Maglasang, Nathan Moran, Gabriel Rolink, and Tyler Samuelson

Logistics

A. Get together with other students on your assigned team in person and virtually.
B. Discuss and complete this assignment in a collaborative manner. Don't just assign different problems to each teammate as that defeats the purpose of team-based learning.
C. Choose a scribe to prepare a final document to submit via Blackboard for grading, changing the file name provided to denote the number of your assigned **Team**.

Problem 1
Consider the logical access control needs for joint software development teams using a typical Linux environment. Roles must include Developers (that can commit changes made in the code), Testers, and Code Reviewers. The technical access control mechanisms that you design must reflect these organizational roles. Your access control solution must:

1. Protect the software being developed from outsiders stealing it
2. Protect against unauthorized changes (including from internal actors)
3. Ensure that we can trace *who* made each change

Answer:
CSCU-Case-Study-Access-Control.pdf

Before we dive into the three scenarios that the case study presents us with, we must first review the parts of this case study. The roles that are present in this case study are the developers; the people that can make changes to or edit the system. The testers will test the system to see if everything runs smoothly, and code reviewers will review the code before it is released to the public. Testers and reviewers will only need read access in all of these scenarios.

The decisions that we made for the technical mechanism will be based on the organization's hierarchy. The hierarchy is put into place to define who has the most authority in the organization. The people that have authority will be able to access intellectual property (IP), and the people that lack authority will be denied access. After explaining the organization's hierarchy, we will present a formal statement mapping out the organization operating system structure, and why the system is structured like this for each scenario.

Our access control suggestions will: protect the intellectual property of the organization, stop unauthorized individuals from invading the or making changes (internally and externally), and keep a log that will record all changes that have been made to the system.

Situation 1: A small team on a single machine *(5 points)*

Answer: since the organization is a small team there are two options to be considered. One option is a classic unix setup. This sort of setup does not require an access control list. All of the developers will be placed into one group. All the developers in this group will be able to make changes to the system. Anyone who is not part of the group will be denied access. The second option is using an access control list(ACL). an ACL will require developers to be added individually to the system. If the company wishes to remove developers that have left the company, they will have to rewrite the ACL. The best option is to use discretionary access controls, users will be able to add and remove people as they see fit. Access can also be given to a specific group of people. This sort of control suits small teams as there are not that many people that need access.

Situation 2: A medium-to-large team on a LAN [Hint: Use of a version control system like Subversion is highly recommended] ***(10 points)***

Answer: A medium to large team will require Non-discretionary access controls. These controls can be implemented by a central authority within the company. Having role-based access controls(RBACs) to achieve the company's needs is also recommended. RBACs allow developers to add roles to each user and classify the importance of intellectual property. Employees who are assigned a role can access all intellectual property that their role has permission to access. If employees leave the team, developers only need to change their role to remove their access. Another approach is implementing a version control system. A version control system can be described as a system where changes are well-recorded by a group of developers. Developers can view who made which change and when was the change made. Subversion, which is a well-known version control system, can be described as open-source access controls. All changes to the system will be filtered through subversion's servers. Using these programs will make assigning access and monitoring changes to intellectual property easy for developers.

Situation 3: A large, distributed team, including outsourced contractors ***(10 points)***

A large organization contains many parts; several teams from different departments, developers and clients. A customer server is a requirement in a large organization. Since this is a large organization, direct access to the version control system is virtually impossible to obtain. This will protect the organization from potential breaches by reducing the amount of people that have access to the controls. Since we are relying on VSC permissions, authentication will be required for all users that access the system. Mandatory access controls(MACs) are recommended for the protection of the system as well. MACs rate each user's level of clearance and the importance of intellectual property. Access will be assigned using these controls.

[Inspired by https://www.cs.columbia.edu/~smb/classes/f09/l08.pdf - many thanks to Columbia University for providing under Creative Commons!]