CIS-481: Introduction to Information Security

InfoSec Chapter Exercise #7

Team: 1

Participants: Zack Graas, Jacob Forcht, Trevor Hagel

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

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

On a single machine we will use an access control list to make three groups and then give each group their desired privileges. The first group developers can log onto the machine to have the software know that they are part of the group. Once they are in, they can have read and write permission so that they can create code for the Linux environment. Also, they can implement a security software that won't allow any outside sources to mess with the code.

Finally, they can use another security system to prevent these outside sources or inside intruders to make changes to the coding for the system. Once they are done with developing the code they can log off and then the Testers can log on.

Once the second group,the testers, are logged in and verified they are assigned read permission and can test to make sure that the code works. Also, they can use fake accounts to test whether any outside accounts can not make changes to the code. Once they are done, they can log out of the machine for the code reviewers.

The last group is code reviewers and they will be granted execution privilege by the ACL so that they can test the code. Also when the code reviewers log in, they can create a software that can trace who made updates to the code, while checking to see if they were updated by someone who works within the group or not. That way, they can fix any changes that an outside/inside intruder has made on the code.

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

In a LAN environment an access control list will be used along with Apache Subversion. The access control list will give permissions based on groups, the groups will be admin, developers, testers, and code reviewers. The access control list will let developers read and write their code so that they can do their job. The ACL will let Testers have execute access so that they can properly test the code. The ACL will give reviewers read access so that they can review the code. Apache Subversion will be used as a version control system on our lan to monitor every change to the code done by each person. The ACL will grant only admins access to the master repository of the VCS, whilst the other employees will have access to a working copy where developers can comment their changes. The ACL will make it harder for outsiders to get access to our LAN and the VCS will record any changes made so in the event of an outsider stealing code we will be able to see who copied the code. Our ACL will help limit unauthorized changes because only authorized groups will have their desired privileges. The VCS will also track who made which changes which can make employees deterred from making malicious unauthorized changes. The VCS will help make sure that each user's changes will be tracked and recorded on the master repository. To help keep the master repository safe the ACL will limit access to it to only admins to reduce the chances of a bad actor hiding changes.

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

For a large and distributed team, access is more important than ever. If each role cannot access what they need to, then work slows down completely. With the access control list roles will be created for each group: admin, developer, tester, and reviewer. Each group will only have access to what they need, while admins will have access to it all. The source code will be held on a remote server and all work will be done on different remote servers. Each of these servers will require a VPN to connect to, limiting outside connections. The servers will run several VM's that a user can connect to to work from. To connect to the VM they will need the proper credentials, thus adding another safeguard from outside connections.

These VM's will have access to source code stored on the remote servers. The source code will have three separate repositories: dev, QA, and production. Production code can only be accessed by an admin(or senior engineer). Dev repos can be accessed by developers and admins while QA can be accessed by reviewers and admins. This system will use Git to manage the source code. In Git, you can use commands to see file change history, enabling tracking of work and commits. In order for code in Dev to be moved to QA, an admin must merge the branch into a similar working branch in the QA repository. There it will be reviewed and once the reviewer signs off as well as the admin, this code can be merged with the master branch of the production repository. The production repository will be locked so it cannot be changed or merged without an admin signing off on a reviewed branch being merged into production. This system allows for in depth review as well as protection for the production version of the source code. No unauthorized changes will be made to source, while any unauthorized changes in QA or Dev can be tracked and rolled back quickly.