HW-5

Graded

Student

Sivachandran P

Total Points

45 / 50 pts

Question 1

Application Isolation

10 / 10 pts

✔ **+ 10 pts Correct**

+ 7.5 pts Mostly Correct

+ 5 pts Partially Correct

+ 2.5 pts Attempt

+ 0 pts Not Relevant/Plagiarism/Use of AI Tools

Question 2

Communicating Applications

10 / 10 pts

✔ **+ 10 pts Correct**

+ 7.5 pts Mostly Correct

+ 5 pts Partially Correct

+ 2.5 pts Attempt

+ 0 pts Not Relevant/Plagiarism/Use of AI Tools

Question 3

Security Problems

10 / 10 pts

✔ **+ 10 pts** Correct

+ 0 pts Incorrect

Question 4

SELinux I

5 / 10 pts

+ 10 pts Correct

+ 7.5 pts Mostly Correct

✔ **+ 5 pts Partially Correct**

+ 2.5 pts Attempt

+ 0 pts Not Relevant/Plagiarism/Use of AI Tools

your solution needs to be succint.(non relevant solution)

Question 5

SELinux II

10 / 10 pts

✔ **+ 10 pts** **Correct**

+ 7.5 pts **Mostly Correct**

+ 5 pts **Partially Correct**

+ 2.5 pts **Attempt**

+ 0 pts **Not Relevant/Plagiarism/Use of AI Tools**

Q1 Application Isolation
10 Points

Web applications belonging to different origins are isolated in the browser, and android apps are also isolated from each other. Enumerate 3 important differences between web application isolation and android app isolation. (Each point you make should be explained within 1-2 sentences.)

Implementation:

Web application isolation - It relies on browser-based sandboxes and cross-origin policies.

Android application isolation - Leverages OS-level features like UIDs, SELinux, and OS manages the granular permissions.

Permissions:

Web application isolation - access is restricted to the resources exposed by the browser application interfaces, and browser restricts direct access of file system.

Android application isolation - Package/Permission manager enforces at OS level through, it ensures granular control in each applications access to system file resources and HW.

Security:

Web application isolation - Content Security Policy (CSP) & Cross Origin Resource Sharing (CORS) policies add layers of security, controls the cross-origin interaction limitations and fetching of resources

Android application isolation - Have access only permitted data and ensured by OS, data flow control by content providers, bound services, Intents, Binder and AIDL (Android Interface Definition Language)

Q2 Communicating Applications
10 Points

What is the mechanism for android inter-application communication? Explain the role of reference monitor in ensuring the mechanism does not violate app confinement as required for security and privacy? (Please be very brief, specific and to the point).

Mechanism for android inter-application communication:

Intent: Its a messaging object used to request an action from App component. Three major uses are Start activity, Start service and send broadcast.

Content provider: Required only when need to share data between multiple applications. It stores and provide content to applications like relation database.

Shared preference: Collection of key values to save, and use the Shared preferences APIs. Applications can use shared preferences to exchange information

External storage: Supports shared external storage space, used to save files. Saved are world-reachable, so applications can use it share information.

Role of reference monitor in ensuring mechanism does not violate apps as required for Security and Privacy:

Enforcing permissions: Checks at runtime to ensure permissions before allowing access to resources or communication channels.

Component access verification and Intent Resolution: While application tries to communicate with another application through an Intent, the reference monitor verifies the intent's permissions and flags.

Restriction of unauthorized data share via content providers: Helps to ensure only authorized applications can access (read and write permissions) specific types of data.

Security-enhanced Linux (SELinux): At kernel level applying MAC (Mandatory Access Control) polices that restrict IPC calls that do not meet security criteria.

Android 6.0 (Marshmallow): Ensures permissions more granularly and permissions are granted at runtime rather than during installation.

Q3 Security Problems
10 Points

Android implements application isolation, and permi

|____|ssion mechanism for mandatory access control implementation. However, one can still have security/privacy attacks on android. Enumerate at least 3 different ways an Android platform may be subject to cyber-attack irrespective of MAC and isolation mechanism.
(Each point you make should be explained in 1-2 sentences.)

Q4 SELinux I
10 Points

In regular unix/linux, processes are naturally isolated as ach process has its own isolated virtual address space. However, even then, if a malicious application is run on regular unix/linux, other applications can be compromised. This is because of DAC implementation in regular unix/linux. Explain how MAC implementation in SELinux may solve this problem. (Please be very succinct and to the point.)

Under standard Unix/Linux DAC (Discretionary Access Control), an application or process running as a user (UID or SUID) has the user's permissions to object such as files, sockets and other process.
With this criteria, in case malicious application running under DAC alone system, could easily exploit vulnerabilities to access or update the files or resources owned by other users or system processes, will lead to potential compromise of enter system

Sharing below some of the limitations of DAC system:

No protection against malicious software
Only two major categories of users: User & superuser
Decisions are only based on user identity and ownership
Each user has complete discretion over his objects
Many system services and privileged programs must run as superuser

How above issues are overcome by MAC (Mandatory Access Control):
---
1) Access to objects controlled by policy administrator
2) User/Process may not change access policy
3) All access are mediated with respect to the policy
4) protects information from:
    - Legitimate users with limited authorization
    - Authorized users unwittingly using malicious applications
5) Provides strong separation of applications:
    - Permits safe execution of untrustworthy applications
    - Limits scope of potential damage due to penetration of applications
    - Functional uses: Isolated testing environments or insulated development environments
6) Provides critical support for application security:
    - Protects against tampering with secured application
    - Protects against bypass of secured applications

With the above listed benefits, MAC helps to solve the problem of exploiting

vulnerabilities via malicious applications. However SELinux is not antivirus software, a replacement for passwords, firewalls or other security systems, an all-in-one security solution.

Q5 SELinux II
10 Points

So far, in the module CS964, you have been introduced to various software coding bugs that lead to vulnerabilities in applications. Such vulnerabilities can be exploited by malicious actors to hijack the control of applications. If the applications are designed to run at high privilege, then exploiting the application can lead to privilege escalation compromising the root. In order for an attacker to exploit vulnerability in an application, either the application has to be network facing, or the attacker must somehow be able to execute another application on the same platform which in turn will exploit a vulnerable application on the platform. So, application confinement and isolation are important to avoid this kind of situation. Most operating systems provide some process level isolation between applications but fail to isolate the applications properly due to improper access control mechanism. SELinux was designed to address this issue. Do a bit of research on SELinux and give a short opinion as to why SELinux is not used more commonly in all linux based servers despite clear security advantages.

Listing below some of the points I have captured:
1) Only some of the appropriate utilities and documentation are installed by default
2) Because often extreme security feature become a pain
3) Policies themselves are sort of magical and complicated, administrative overhead
4) Whenever SELinux blocks access, it issues an error message, but these messages are often very vague, which makes trouble shooting rather difficult
5) Policies sometimes be overly restrictive
6) Lading to compatibility issues where legitimate applications or services are blocked, creating operational friction
7) The steep learning curve and the risk of inadvertently locking down essential services often leads administrators to either disable SELinux or opt for simpler, more manageable security solutions, even if they are less robust.