Question 3

a) A Unix based implementation of the Tetris game maintains information on player scores in the file /etc/scores. The game is SUID root and is executable by all, the scores file is owned by root and is readable and writable only by its owner.

i. Why is the game SUID root? Explain how the SUID permission operates. (7 marks)

- So that for any user playing the game, the program can update the highscores file with the player's score.

- When a program file has the setuid root permission set then during execution the user id of the invoking process becomes root.

ii. Discuss the dangers of using SUID root and suggest a safer way of securing the program and file. (8 marks)

- SUID is dangerous because any user can run the program with "root" privileges ie. the user will essentially have access to the etc/passwd file. Any program granted SUID privileges must be thoroughly inspected and totally trusted. The scripts must be readable by the user who executes them, so any small bug in the script can be exploited, and this exploit in turn may be used to compromise the system.
- I would suggest using a Type Enforcement Policy as a safer way of securing the program and file.

b) Describe the Type Enforcement protection model. Using the Tetris program above as an example, explain how and why Type Enforcement can provide stronger protection than a SUID based mechanism. (15 marks)

- In a Type Enforcement Policy, every subject (process) has an associated protection domain. Domains entered by executing a program attached to that domain. Domains are like sandboxes that are used to limit the access that a program has to resources.
- As mentioned above, the domains limit the access a program has to resources, meaning that the process running the program has limited privileges ie. the user only has the most basic requirements to run the program and no more.
- This limiting of the users privileges ensures that the user does not have any extra permissions that they do not need, which may in turn be used for malicious means.
- So even if a program is running as SUID root, as long as it is in a protections domain with limited privileges, the attacker will have limited access to other system services etc. which are within other protection domains.

c) The SUID root Tetris program described Part (a) above may take as parameter a path to a scores file (to override default /etc/scores path). This program has behaviour:

```
void main (int argc, char* argv[]){
        char scores[12];
        strcpy(scores, argv[0]);
        // argv[0] gives path to scores file
        ...// Step 0. play game;
        ...// Step 1. open scores file to obtain user's last score;
        ...// Step 2. create/open temporary file stmp in same directory as scores;
        ...// Step 3. open scores file, copy contents to stmp and current score;
        ...// Step 4. close files and rename stmp as score file;
}
```

Identify and explain potential security vulnerabilities in this design. (15 marks)

- This design is open to a buffer overflow and stack smashing attack.

- Answered in Summer 2011