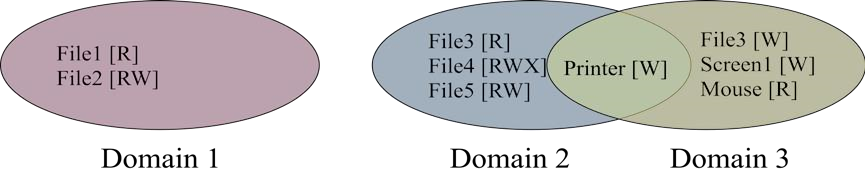
**Operating systems, security and networks (207SE) Lab 19: Security**

In this portfolio activity we will consider the issues of protection and security in operating system by considering approaches to control access to resources, password systems and encryption.

**Activity 1**

1. Create a protection domain matrix, access list and capability list for the diagram below.



|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Protection domain matrix | | | | | |  |  | | |
|  | File1 | File2 | File3 | File4 | File5 | Printer | Screen1 | Mouse |  |
| D1 | R | RW |  |  |  |  |  |  |
| D2 |  |  | R | RWX | RW | W |  |  |
| D3 |  |  | W |  |  | W | W | R |
|  | | | |  |  |  |  | | |

**Access List (column)**

File 1

Domain 1[R]

File 2

Domain 1[RW]

File 3

Domain 2[R]

Domain 3[W]

File 4

Domain 2[RWX]

File 5

Domain 2[RW]

Printer

Domain 2[W]

Domain 3[W]

Screeen1

Domain 3[W]

Mouse

Domain 3[R]

File 3 : [W]

Printer : [W]

Screen1 : [W]

Mouse: [R]

File 3 : [R]

File 4 : [RWX]

File 5 : [RW]

Printer : [W]

File 1 : [R]

File 2 : [RW]

**Capability List**

1. Create a basic salted-hash function in a programming language of your choice that takes a password as a string and produces a pseudo-random integer based on that string. You could devise your own approach to create this function or you could
2. take each letter of the string and converting it to its ascii value,
3. multiple it by a specific number based on its position in the string sequence, (iii) add the values together and add salt in such as the current date.

**Lab Activity 19 Security**

1. Create a protection domain matrix, access list and capability list for the diagram below

[Diagrams here]

1. Commented Code showing hash function

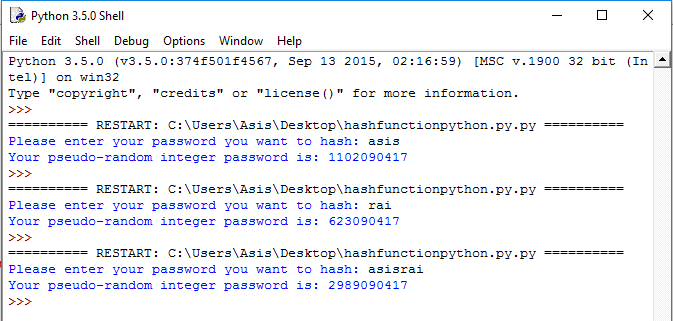
Program used to create hash function: Python

**Python Code**

[Commented code here]

1. """a basic salted-hash function in a programming language of your
2. choice that takes a password as a string and produces a pseudo-random
3. integer based on that string"""
5. **import** time
7. #input
8. password=input("Please enter your password you want to hash: ")
9. #putting password in a list
10. password=list(password)
11. #new list
12. hashpassword=[]
14. #asciivalue function, take each letter of the string and converting it to its ascii value
15. **def** asciivalue():
16. **for** i **in** password:
17. j=ord(i)
18. hashpassword.append(j)
19. **return**(hashpassword)
21. #multiply function, multiple it by a specific number based on its position in the string sequence
22. **def** multiply(hashpassword):
23. Counter=1
24. integer=0
25. **while** integer<len(hashpassword):
26. hashpassword[integer]=hashpassword[integer]\*Counter
27. Counter=Counter+1
28. integer=integer+1
29. **return**(hashpassword)
31. #add and salt function, add the values together and add salt in such as the current date.
32. **def** addnumberandsalt(hashpassword):
33. total=sum(hashpassword)
34. total=str(total)+time.strftime("%d%m%y")
35. **return**(total)
37. #run functions
38. asciivalue()
39. multiply(hashpassword)
40. total=addnumberandsalt(hashpassword)
41. **print**("Your pseudo-random integer password is: " + total)

**Output showing your salted- hash function working**



# Marks

Activity 1 – 2 marks

Activity 1 and 2 – 5 marks

# Evidence

Protection domain matrix, access list and capability list

Commented code and output showing your salted- hash function working

2