28/9/18

① ?

Better way of doing : sleep till semaphore is unavailable

```
wait
~~while~~ (s)
{
  s --;
   if (s<0)          → as soon as s =1, it'll get up.
     sleep (S);
             ↳ can be T
}
```

Suppose s = -~~5~~ 3 → ~~5~~ 3 processes waiting
1 process finishes → s = -2
Another → s = -1
As s=1 ⇒ it will wake up.

```
P:  wait(s);
      //CS
    signal (T);
```

Suppose : P : Q:

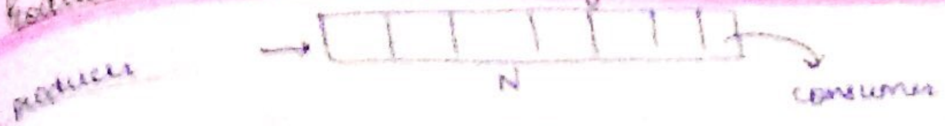
pf ("0") pf ("1")

Cond^n : 1st 0 ~~printed~~, then 1 should be printed

↓

```
s = 1                 T = 0
P :                   Q :
wait (s);             wait (T);
pf ("0");             pf ("1");
signal(T);            signal (S);   → serialize me
   ↓                                   use :→ wait &
want Q to                                    signal
~~run now. T=0~~
so, signal (T)
```

Producer - Consumer (Signalling Mechanism)



Initially, count = 0. As P puts something → count++;
C consumes → count--;

```
producer () {
    ~~produce (item)~~     if (count == N-1) {        // wait
    ~~count++;~~                 sleep (cv1)
                       else
                           produce (item).
                           Count ++;       :→ if count ↑, I should
                                              inform consumer
                           wakeup (cv);
}

consumer () {
{
  if (count == 0) {
      sleep (cv);
  else
    consume (item);
     count --;
    wakeup (cv1);                          // wakeup the producer
}
```

Protection & Security

Protection

↳ Protection : control access to system

↳ which user can use which prg.

↳ mechanism for enforcement of policies governing resource use

resource ↓

CPU, Memory, Printer, secondary mem ← H/W Objects

+ S/W Objects

Files, Processes, ~~semp~~-semaphores.. ←

if someone uses something not allowed ; policy how to deal with him

Policy ⟶ Mechanism

what needs to be done — How to do it

If process is trying to access another's add. space
policy : not allowed, error should be sent, trap to OS

Access : Each object has an access right associated with it.
(what op's you can perform on it)

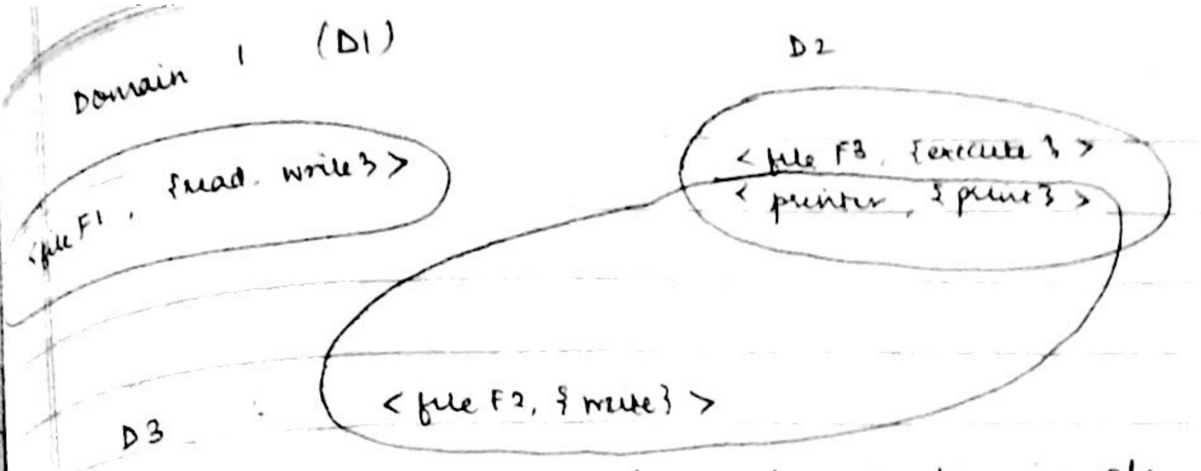
Domain :

< object , access - right >

eg. < file F1 , { read, write } >
< Printer , { print } > } ← domain

↳ work on principle of least privilege
A process should be given only those rights which it needs to complete its current task
(min. rights)

Teacher's Signature



↳ can be overlapping also

↳ let : U1 : work only in D1. > can only access F1 in RW mode
> can't use other resources

↳ P2 : works in D3 ⇒ can only access $\overset{\text{W mode}}{\text{F2}}$ & $\overset{\text{print mode}}{\text{printer}}$

"Need-to-know" Principle : Only domain should be given to P/User & which is min^m required by him at that time

Allocate Resources (Assigning D)

Static

D will remain with you throughout (which given to you in starting)

suppose need D1 → $\overset{1st}{10ms}$

D2 → $\overset{next}{10ms}$

But I'm giving all in starting

↓

violating Need to know Principle

Dynamic

When you need one, I'll give you that Domain

* All this info is kept in (Domain) Access Matrix

Proc 1

Objects / Domain	F1	F2	P1	D1	D2	D3
D1	read	write				
D2	read	read execute	owner	control		switch
D3	read*		execute		switch	
D4			execute			

can add col if needed

→ If I'm in D2, I can switch to D3

→ access ~~ri~~ right on an object in particular domain

→ D2 can control D1.

↳ can specify domains

↳ Here, your permission is checked.

↳ Dynamic switching :→ may want to switch to ~~D~~ D3 (to execute F1)

⇓

domains are also objects

access (i, j) = switch ⇒ If I am in Domain Di, then I can switch to Domain Dj.

→ Suppose I'm in Domain D3

read* ⟶ allows me to copy my access right in that column only.

⇓

I can give 'read' access right to any domain for F1

↳ ~~[illegible]~~ transfer copy rights : give to someone else but I'll loose right

↳ Owner : who has all rights to Proc 1

↳ can give rights to diff domains of Proc 1 (other or same)

Teacher's Signature

Control : D2 can control D1
→ D2 can change access rights, modify D1.

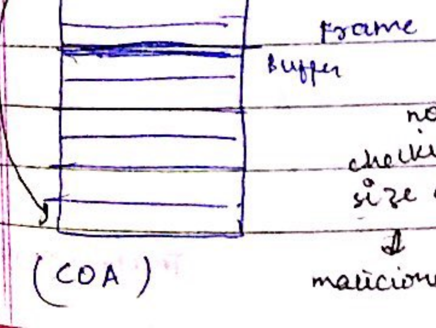
Security

Protecⁿ
taking care of system.
~~Att~~ Multiple user :
how to control

Syst Securities :
Protect from external environment
(hacker, intruder)
Threat : Possibility system may be ~~attacked~~ harmed
Attack : Someone is harming your system.

Breach of Confidentiality : unauthorized reading of data
Breach of integrity : " " ~~identification~~ modification
" " availability : " destruction
Theft of service : " " use of resources
Denial of service (DOS) : preventing legitimate use of service
{ not allowing even those to use resources who're allowed }

Stack and Buffer Overflow
Everytime a funcⁿ is called, a frame is created



```
main ( )                → capacity : 9
{   char buf [10];
    char s;
    sf (" %s", &str);
    strcpy ( buf, s)       → not even checking size of str
}
```

(COA)

malicious user : gave 20 add.

I've put : malicious code commands

↓ converted into

binary

↓

all stored in stack 4 gave return add. at bottom of stack.

↓

by debugging so much time so that I know where is return add. & what is bottom of stack add

→ pretend to be someone else & get info. :→

↓ to avoid

use cryptography
(encrypt your msg)

K - set of keys

M - " msgs

C - ciphertext (new msg after encryption)

E : $K \xrightarrow{s.t.} (M \rightarrow C)$

encrypt — using key — msg become cipher text

D : $K \xrightarrow{s.t.} (C \rightarrow M)$

Decrypt

Encryption

Symmetric — Asymmetric

S & R Both are having same keys ← Symmetric

1 K only
↓
used for Encryption 4 Decryption

Teacher's Signature

1) Symmetric

↳ Same key will be used to encrypt & decrypt

↳ If this key is compromised (someone else knows this key) → someone else may know msg

↳ msg is gone

DES (Data encryption std.) :

takes 64-bit value

apply 56-bit key

R → also has same key to decrypt

This applied on block. : Block cipher
(64-bit, ...)

↳ If I keep on checking msgs, $\overset{\text{intruder}}{\text{I}}$ can very soon figure out key being used.
can use

Triple DES : Using 3 keys

E(K1)(m) : Encrypt msg with 1st key K1

D (K_2) : decrypt using key K_2

E (K_3) ( ) : send to you

$$E(K_3)\left(D(K_2)\left(E(K1)(m)\right)\right)$$

Better version

Advanced Encry. std. (AES)

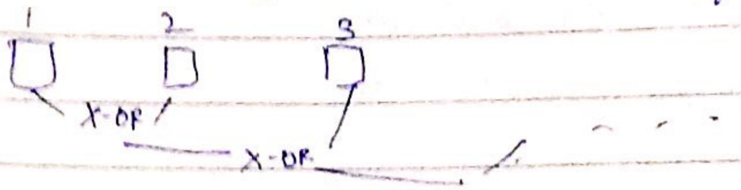
Uses 128, 192, 256 -bit keys rather than 64-bit key

↳ larger keys & blocks

Apply key on 1st & 2nd Block. Then, EXOR the blocks

↓

(keep on X-OR ing on subsequent blocks) → Block chaining (forms chain)



↳ Becomes difficult for intruder to find out

↳ Apply key of same length

→ Problem : key if is compromised → msg is gone

2.) Asymmetric

RSA (Algo most commonly used) :

Telling everyone which key is used for encryption

↓

Public key

Asymmetric ← (K_e → Public Key. (everyone knows)

K_d → key for decryption : Only 1 person knows (Reciever)

Private key

Encrypt :→ $\boxed{E(Ke, N) = m^{Ke} \bmod N = c}$ cipher text

Decrypt :→ $\boxed{D(K_d, N) = c^{Kd} \bmod N}$

$N = p \times q$

↳ very large prime no. (512 bit each)

This cond^n should be satisfied :

$(K_e K_d) \bmod ((p-1)(q-1)) = 1$

↳ Suppose $p = 7$, $q = 13$ $\qquad (p-1)(q-1) = 6 \times 12 = 72$

$N = 91$

1 choose $K_e = 5$ → prime $\qquad$ gcd = 1

$5 \times K_d \ \% \ 72 = 1$ $\qquad \frac{145}{5} = 29$

$\Rightarrow 5 \times K_d = (72 \times i) + 1$

$\Rightarrow \boxed{K_d = 29}$

$E(5, 91)$

$D(29, 91)$

have to : $E(69) = (69^5) \% 91 \Rightarrow 62$

decrypt $D(62) = 62^{29} \% 91 = 69$

Ques Alice generates RSA keys by selecting $p = 11$ & $q = 13$. She chooses 7 for her RSA public key. (K_e). Bob wants to send plain text msg no. 9 to Alice.

$N = 11 \times 13 = 143$

$(7 \times (K_d)) \% (120) = 1$

120 $\quad$ 149

240

360 $\quad$ 121

480