PROGRAM SECURITY

- Program Security
- Non Malicious Errors (No bad Intentions)
- Malicious Attacks (Bad Intentions) e.g:- Virus
- Targeted Malicious Attacks. (Target a particular system)
- Controls.

TO DETECT WHETHER A PROGRAM IS SECURE OR NOT :-

1] PENETRATE AND PATCH METHOD:-

CRITERIA :- No of faults.

faults → 20 faults → 50

patch 20 patch 50

2] UNEXPECTED BEHAVIOUR:-

secure { Desired Behaviour ⇒ Properly Performed

Undesired Behaviour ⇒ is eliminated / avoided.

Program flaws

Inadvertent human error — Malicious, intentionally induced flaw

Known flaws fall in 8 category

① Validation error ② Domain error ③ Serialization & Aliasing

④ Inadequate identification & authentication

⑤ Boundary Condition violation ⑥ Other exploitation logic errors.

NON MALICIOUS ERRORS :- (No bad Intentions)

(Honest people making honest mistakes)

1] BUFFER OVERFLOW :- (DATA SIZE)

define size of buffer

```
char Sample [10]
```

unknowingly

detected during compilation → `Sample [10] = 'B'` ← Buffer overflow

detected during execution → `for (i=0, i≤10, i++) sample [i] = 'A'` ← weakness vulnerable if i = 10

BUFFER OVERFLOW IS INTO :-

overwrite

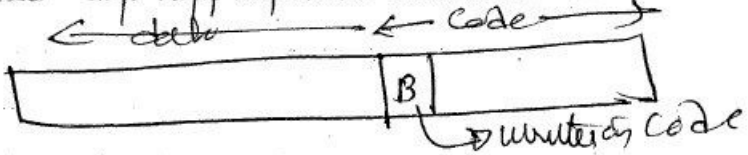
user data

Only one process is affected.

1] USER DATA :-

Only that data item is affected / overlapped.
It can be detected by appropriate control.

2] USER CODE :-



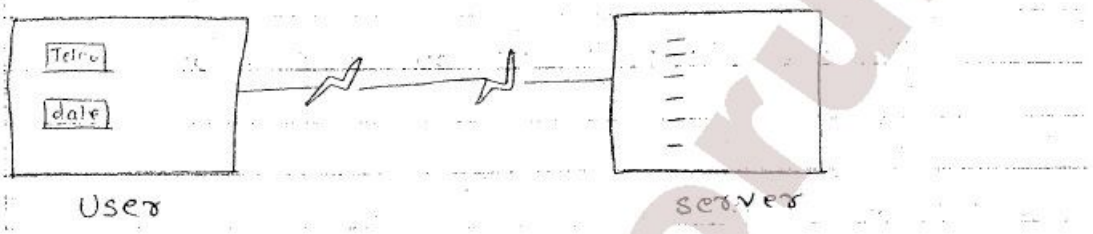
- Error :- Exception 6 ⇒ Invalid instruction (stops execution)
- Unexpected behaviour

3] O.S. CODE / O.S. DATA :-

Entire system gets affected

ATTACKS :-

http:\www.mysite.com / userpage/ & param_1 = (022) - 2379586 & param_2 = 02APR2006.



2] INCOMPLETE MEDIATION :- (DATA TYPE)

eg

www://www.mysite.com/userpage/ &Item_no = 155 & Qty = 10 & unit_cost = 100 & Total = 1000

Data is getting exposed to the user. If in the above example, user sends the unit cost as 10 instead of 100, it will be loss for the company

5] TIME OF CHECK to TIME OF USE :-

Access control: only intended user should be able to access an object.

(SERIALIZATION PROBLEM "bait & switch")

Time Gap between two sequential instruction :-

- Multitasking.
- Out of order execution. (Pentium)

Token - Ticket to o.s.

User X	Access X

→ Encrypt this token for checking & decrypt the same to access the time.

User Y	Access Y

In case of accessing the system, the system may check user X token to check proper system & Access Y token to accessing that system because of some time intervals while checking. the tokens.)

Malicious Attacks

(Done with bad Intentions)

TYPES OF MALICIOUS ATTACKS :-

1 VIRUS :-

Virus is harmful when it gets executed or triggered by some event.

i) TIME BOMB :-

Virus is harmful when it gets triggered by time related event.

eg :- "happy birthday Mr Joshi"

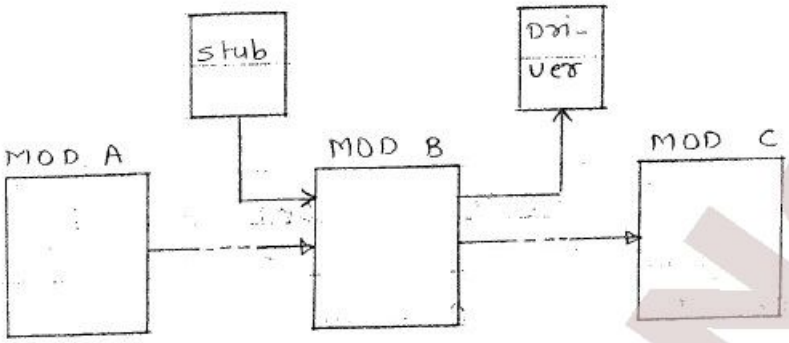
⇧ This virus gets executed & harmful at the time of birthday of Mr. Joshi.

ii) LOGICAL BOMB :-

Virus is harmful when it gets executed or triggered by some event.

iii) TRAP - DOORS :- (stub / driver)
i/p o/p

Trap doors are undocumented entry points into the program. Also called as back-doors / secret doors.



1] Unintentionally forgot to close stub & drivers.

The above example tells about stub & driver. If mod B requires o/p from A but mod A is still in progress then B will make its own i/p that is stub to check the system whether it is working properly or not

2] Good Intentions :-
- Maintainance
- Patch/ upgradation

3] Bad Intentions.

iv] TROJON HORSE :-

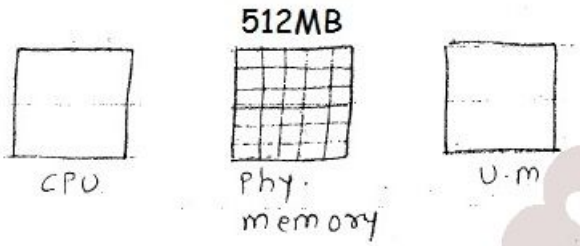
To enter into the system secretly to destroy it is called as Trojon Horse attack.

v] WORM :-

It replicates itself using the network connection.

vi] RABBIT:-

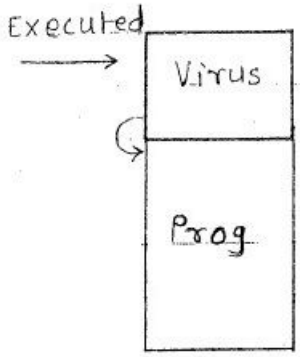
It creates its own multiple coppies because of which some system resources can be made unavailable.



If Rabbit is of 1 mbyte in size it will make 512 mbyte copies of itself & copies in the physical memory & makes that physical memory unavailable to user since it is exhaustive.

HOW VIRUS GETS ATTACHED :-

1] ATTACHED WITH PROGRAM :-

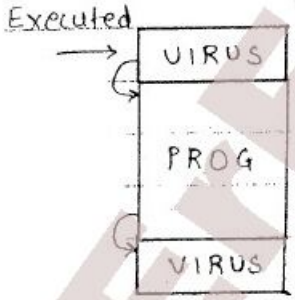


⇒ Actual size of prog is known
eg:- 378 Kb

⇒ Detection is easily possible

⇒ Removal is easily possible.

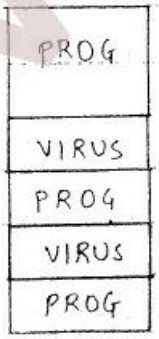
2] SURROUNDED WITH PROGRAM :-



⇒ Detection is easily possible.

⇒ Removal is not easy

3] INTEGRATED WITH PROGRAM :-



⇒ Useless Instruction

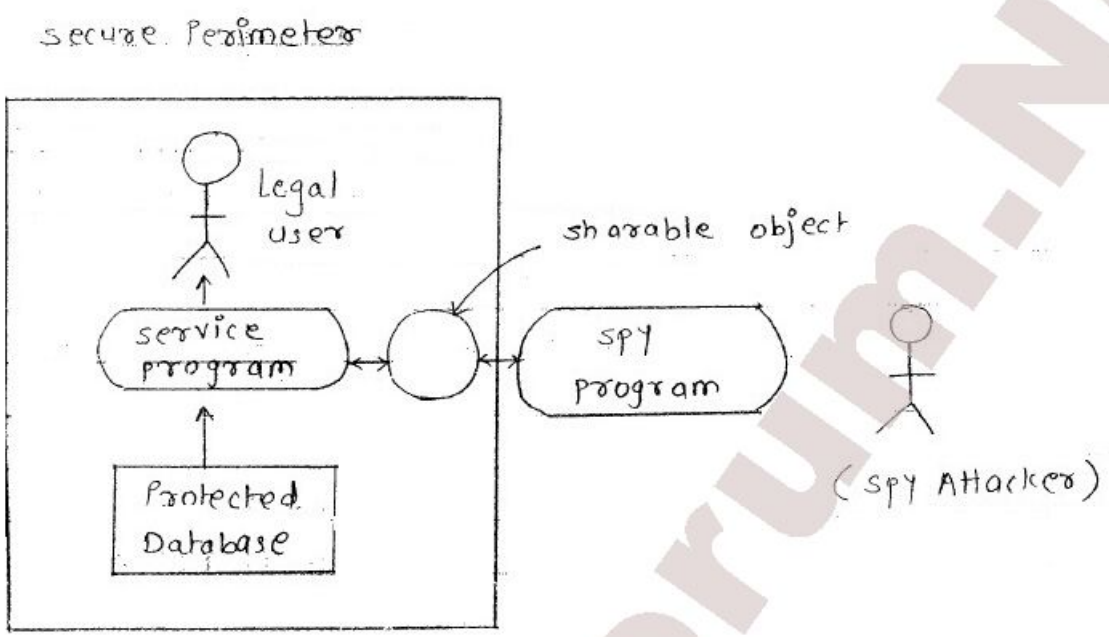
⇒ Jump to next sequential Instruction

⇒ Add 0 to same no.

⇒ Detection is not easily possible.

Targated Malicious Attacks:

Covert Channel= channels that leak information
(open channels)



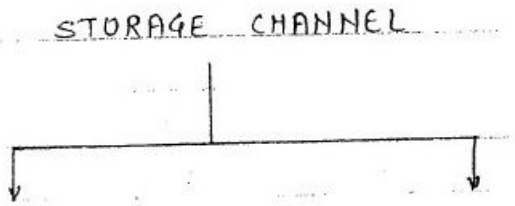
Service program is written by spy programmer. He is not having any access to that program after delivering it. Only legal user can use the database through service program.

REQUIREMENTS TO ATTACK IS BEING TAKING PLACE:-

1] sharable resource
2] common clock sense

For common clock sense there will be no problem since every comp has Real Time clock.

COUERT CHANNEL IN TERMS OF STORAGE CHANNE



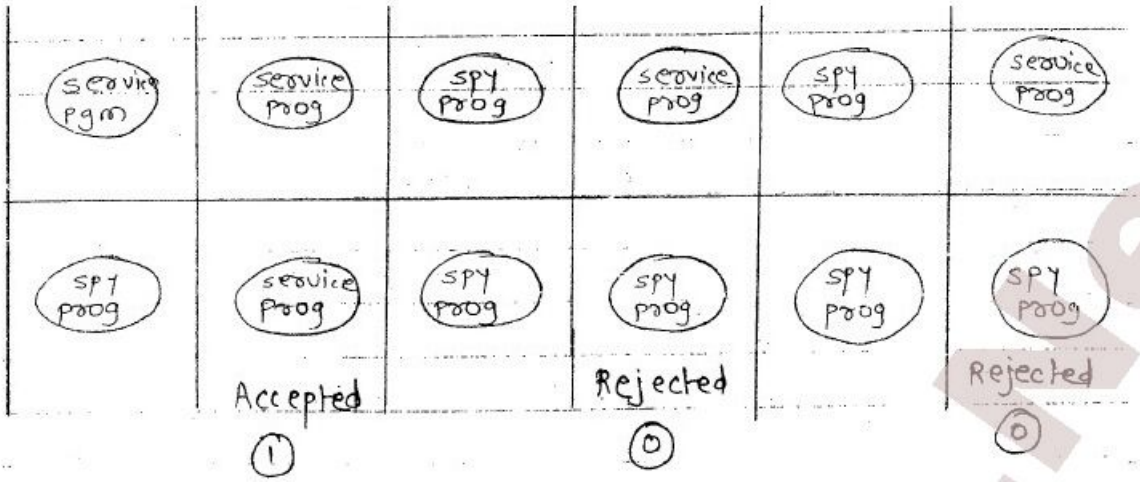
lock = 1	Existance = 1
Unlock = 0	Non Existance = 0

In file elock channel when 1 is to be sent spy program locks the sharable object & if 0 is to be sent it will unlock the sharable object.

In File Existance channel, when 1 is to be sent spy program will create a new directory/ file & if 0 is to be sent it will delete that file.

COUERT CHANNEL IN TERMS OF TIMING CHANNEL

In Hming channel sharable resource is CPU's Hime. The Hime slot is allowed by CPU is accepted for logic 1 whereas rejected for logic 0.



If timing slot alloted by CPU is taken by service program then 1 is to be sent.

If timing slot alloted by CPU is not taken i.e rejected by service program then 0 is to be sent.

DISADVANTAGE :-

Timing channel is not useful in multitasking enviornment. It is useful only if 2 processes are running at the same time.

How to convert Timing channel into storage channel

Timing channel can be converted into storage channel by considering CPU time as sharable object & ensuring as only two process are running in round robin fashion.

DETECTION OF POSSIBLE COVERT CHANNEL:-

1] SHARED RESOURCE MATRIX:-

RESOURCE	SERVICE PROGRAM	SPY PROGRAM
SHARED OBJECT	Read, Modify	Read
PROTECTED DATA	Read	(-)

2] INFORMATION FLOW ANALYSIS:-

B := A $A \rightarrow B$ (Explicit flow)

B := A $A \rightarrow B$, $B \rightarrow C$ } Explicit info flow

C := B $A \rightarrow C$ → Implicit flow

eg:- if D = 1 then B := A

$A \rightarrow B$ – Explicit

$D \rightarrow B$ Implicit. (iff info will from B if D = 1)

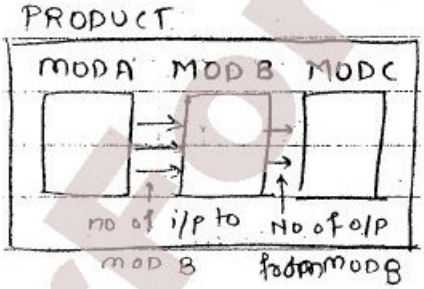
PROGRAM CONTROLS :-

I] DEVELOPMENT CONTROL :-

- Modularity
- Encapsulation
- Information Hiding.

MODULARITY :-

To avoid possibility of attack on application program, the program is always divided into smaller Independent Modules.



Every Module of a product should be :-

a) simple
b) Independent logical Entity
c) small

ENCAPSULATION :-

Encapsulation means, the no of inputs to the module should have well defined

INFORMATION HIDING :-

It means, the no of outputs should be well defined.

II] PEER REVIEWS :-

INFORMAL REVIEW :-

It is a simple review. Artifact i.e. product is presented for review.

WALKTHROUGH :-

Creater of an artifact will only present the product.

INSPECTION :-

Creater do not lead the presentation. He will be just part of a preparation of the product. But someone else will present the product.

III] QUALITY CHECK - TESTING :-

1] UNIT CHECK

Every module of a product will be tested.

2] INTEGRATING CHECK :-

The whole product as a whole will be tested.

3] BLACK BOX TESTING :-

For every i/p the product give proper o/ps.

4] WHITE BOX TESTING :-

The whole program will be tested including i/ps & o/ps.