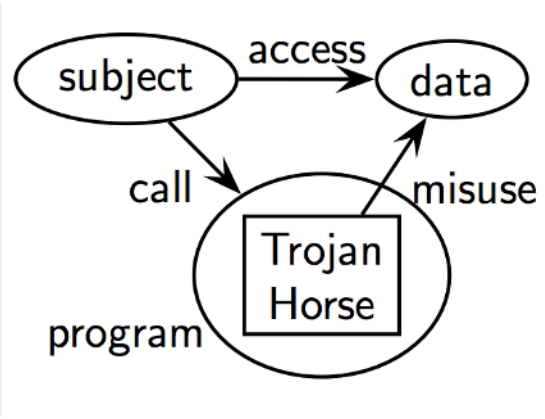
Trojan Horses

/tmp is used for holding temporary or larger files. TH can be found in lots of useful software that contains malicious code that can be later used as an exploit.




```
# Trojan /tmp/ls
#Steal rights
#/bin/sh
chmod a+rwx $HOME
/usr/bin/ls
```

Should an user create the above script and an unsuspecting user with a '.' at the start of their PATH performs an **ls** on the /tmp folder. This will change the permissions on that or any given directory.

An attacker could conceal the attack by hiding the changed files so as the user does not suspect anything.

Does this violate the UNIX security System ?

Well according to the unix policy which is a discretionary policy a user can choose to allow anybody access to all the files they wish. In the eyes of the system the user who has permission to change permission has just changed permission.

This demonstrates how inadequate the system is as it cannot protect us from this kind of attack. In this case the system is not broken, rather our definition of security is ambiguous.

This again point towards what is actually meant when we say security.

THe user (subject) call the program and the trojan misuses the data which that user has access to in the first place.

Malicious code installation.

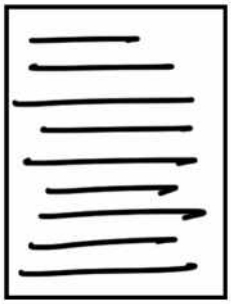
We are always bombarded with popups asking to install things that we often accept and unwittingly accept some javascript that can also be encoded in a pdf and runs silently on our machine.

- Code is not always concealed in programs
- Code can be embedded in documents also
- There are many ways which malicious code can run on our systems.

Software Features

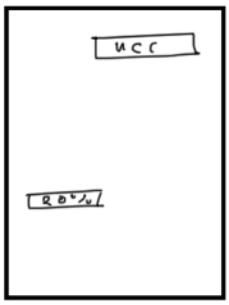
Sometimes software features can provide an exploit for a trojan horse.

Microsoft Fastsave.



In the past the MS Word would read the whole file into memory and make changes and then write the whole file back to disk.

This was when files were smaller and its was also slow to write to disk



The new approach was that the file exists out on disk and any changes would be appended. Only when the file is next opened by another user is the whole file then rewritten.

This was the standard template used by fast save so when the second person opens the file it was possible to discover historical information which was stored in the meta data.

The same was true for redacted documents, anyone with a PDF editor could just go to this meta data and remove the black boxes covering the sensitive data.

[How to redact correctly]

Also if you are going to redact a document then you should be consistent because information may be learned from the fact that what was not deemed sensitive before now appears redacted

Two Types of security (MAC & DAC)

Discretionary Access Control

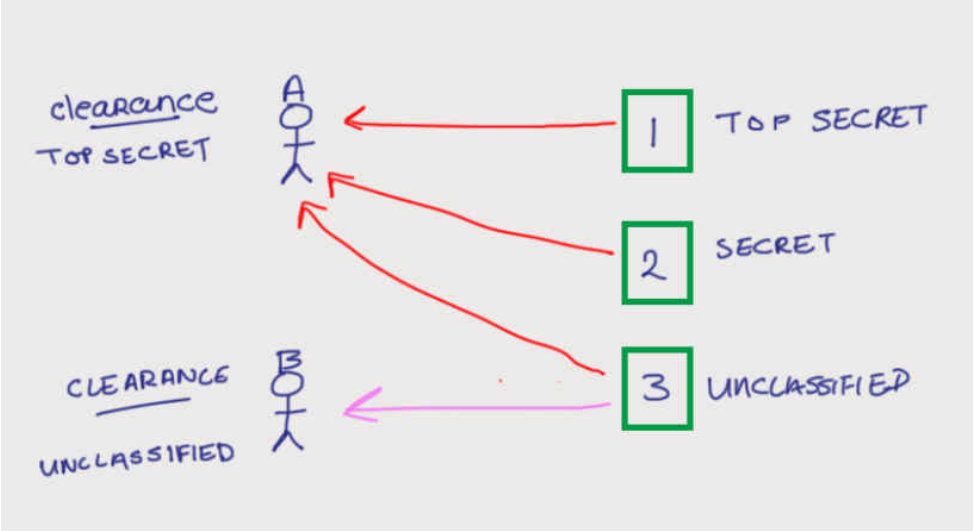
- I have the choice as the user to change the access

Mandatory Access control

- UNIX group access , only system administrator can grant access

Multi Level Security

- Easy to understand but hard to implement correctly

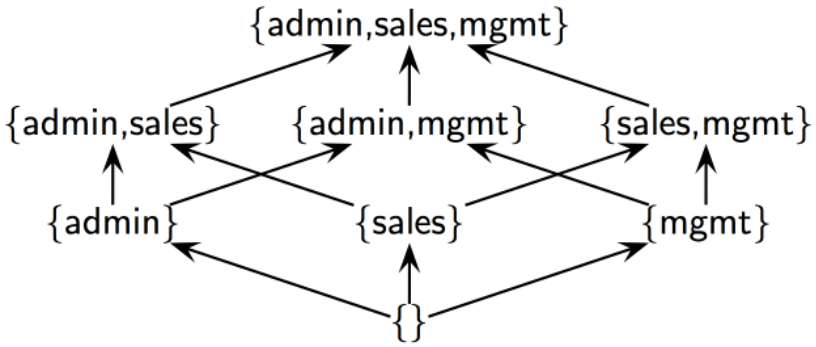


User A is cleared to *Top Secret* and anything lower
User B is cleared to *Classified* and anything lower

If user A writes to or creates a new file at classified and pastes information up to secret ?
This is fine as the information is flowing up.

If user A writes to or creates a new file secret and pastes down to classified ?
This breaks the order as information is not allowed to flow down.

Multilevel Security Classifications



In the above model consider a document S that contains only sales information and has the classification { sales }

Consider a report R which contains both sales and administrative information and has the classification { sales , admin }

It's allowed for the information in the document S to be in the report R since

{ sales } ⊆ { sales, admin }

It's not allowed however for information form the report R to be posted to the document S since

{ sales, admin } is not ⊆ { sales}