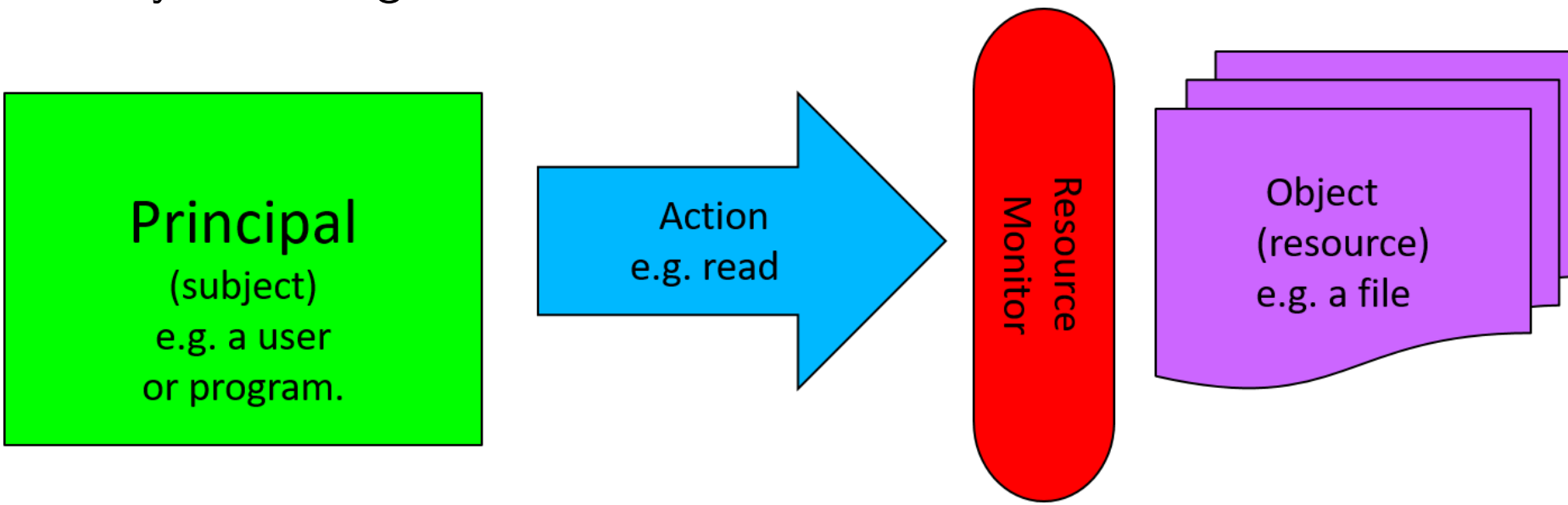
Access Control

Introduction

- Need to ensure that only authorised users have access to what they need
- Will discuss ways of achieving this and possible pitfalls

Model of Access Control

Start by discussing access to data



Access Control Matrix

	Operating System	Accounts Program	Accounting Data	Audit Trial
Alice (manager)				
Bob (auditor)				
Accounts Program				
Sam (sys admin)				

Permission: x: execute, r: read, w: write

Access Control Matrix

	Operating System	Accounts Program	Accounting Data	Audit Trial
Alice (manager)	x	x	-	-
Bob (auditor)	rx	r	r	r
Accounts Program	x	r	rw	w
Sam (sys admin)	rwx	rw	-	-

Permission: x: execute, r: read, w: write

Access Control Matrix

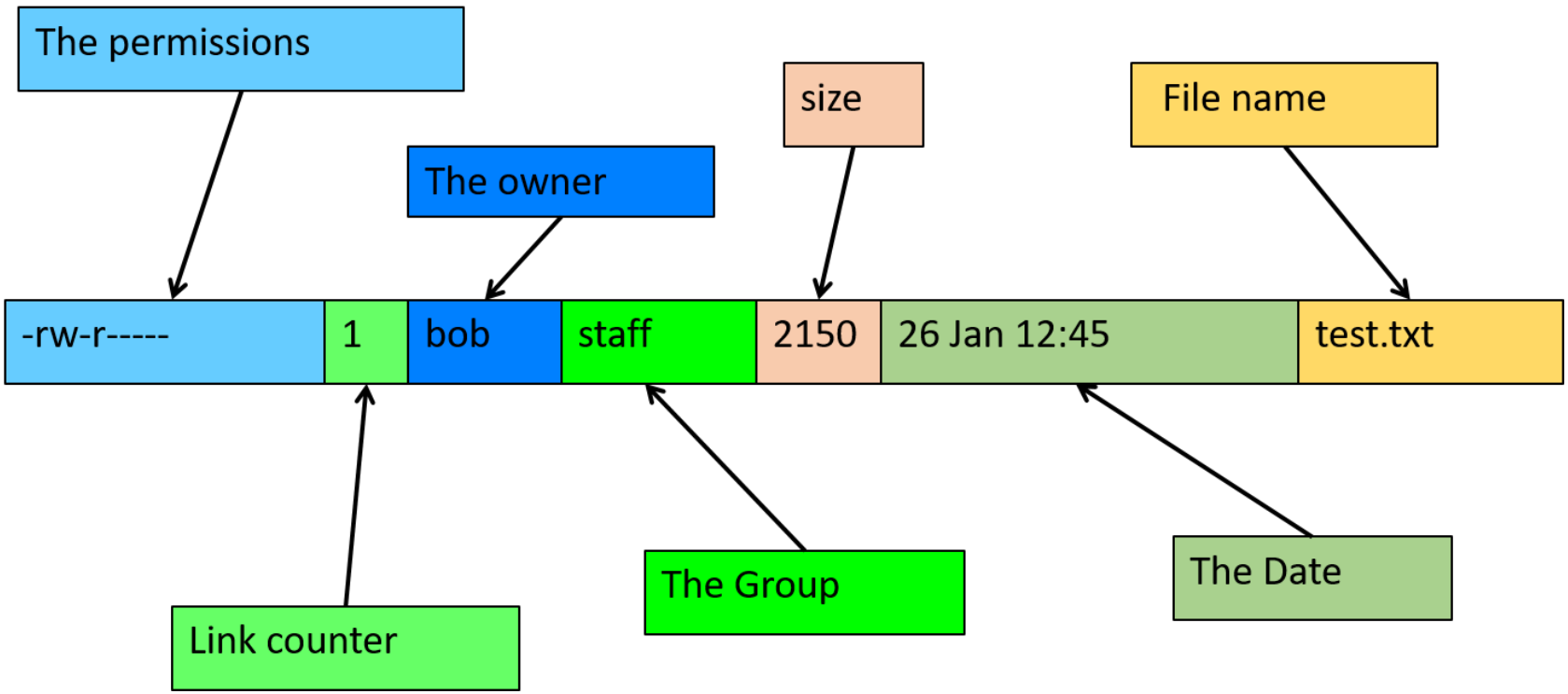
- ACM is a matrix of all principals and objects
- The matrix entries describe the permissions
- Problem: maintaining such a matrix can be difficult
- If the matrix is corrupted, then all control is lost

Access Control Lists (ACLs)

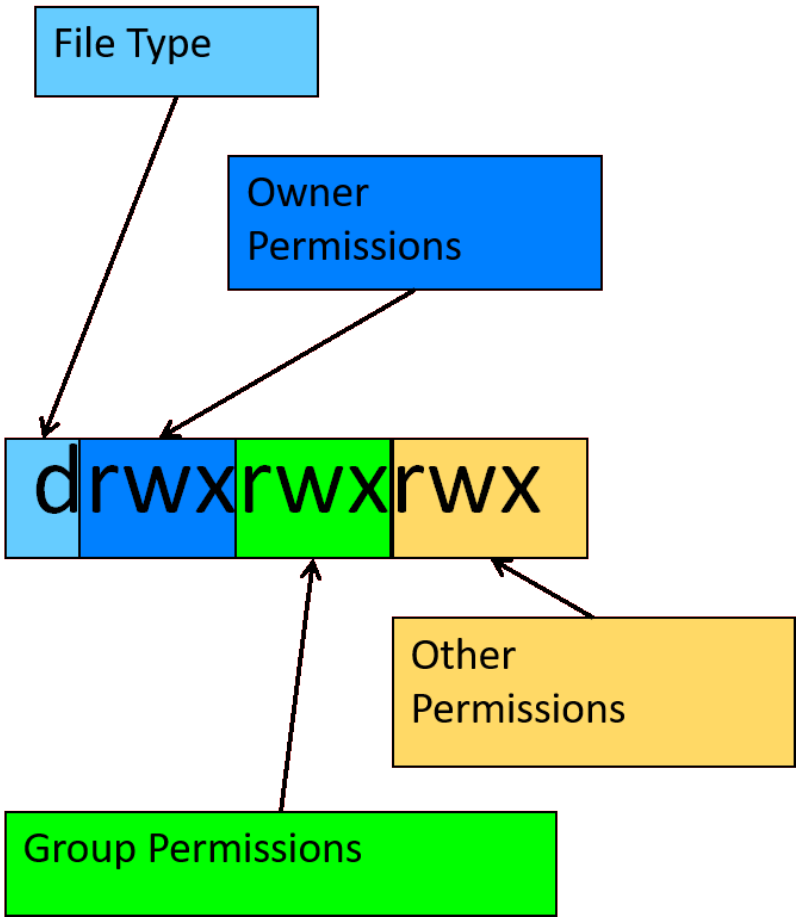
- We don't want to store one massive matrix.
- Instead we can store each column of the matrix with the object it refers to, eg.

  (Accounts data, [(Sam, r), (Bob, r), (Accounts program, rw)])

The UNIX Access Control List



UNIX File Permissions



- Permissions:
  - r: read permission
  - w: write permission
  - x: execution permission
  - -: no permissions
- File Type:
  - - : file
  - d : directory
  - b/c: device file

Access Control for Directories

- For directories:
  - "r" is read only for directory contents
  - "x" is permission to traverse, e.g. switch to, run.
- No "x": I can't run any commands inside the directory
- No "r": I can't list the files in the directory

Access Control for Program

```
-r-sr-xr-x   1 root   wheel 70352 19 Jun  2009 passwd
```

- The “x” permission controls who can run a program in the case of passwd: anyone.
- The “s” permission indicates that the program runs with the permission of its owner.

Different user identifiers

- Have different user identifiers (uids):
  - real uid (`ruid`) owner of process
  - effective uid (`euid`): used for access checks (except filesystem)
  - file system uid (`fsuid`): used for access checks and ownership of files (usually equal to effective uid)
  - saved user uid (`suid`): when the `euid` is changed, the old `euid` is saved as `suid`. Unprivileged process may change `euid` only to `ruid` or `suid`.

  Provides flexibility for granting higher privileges temporarily
  - eg daemons: start as root (to bind to ports < 1024), then set `ruid`, `euid` and `suid` to unprivileged values. Cannot gain root privileges afterwards
  - Process run as privileged user may set `euid` to unprivileged value, then execute non-privileged operations, and gain root privileges afterwards

Security issues with granting higher privileges

- Users can run process with more privileges
- If there was a mistake in the passwd program we could use it do root only actions.
- Particular problem: race conditions in code like

  ```
  if can_access file then perform_operations on file
  ```

- **Make sure process have as low a level as possible.**

Storing Passwords

- Passwords not stored in clear text
- Only hashes are stored
- Further security measure: Store pair `(Salt, Hash)`, where `Salt` is random bitstring, and `Hash` the hash of the salt and the password
- $\Rightarrow$ Same password for two users gives rise to different entries in the password file
- Makes cracking passwords much harder

Windows Password Hashes

- Windows stores its password hashes in: `system32/config/SAM`
- This file requires Admin level to read.
- It is locked and encrypted with a key, based on other key values.
  This adds no real security

Password Hashes in Windows Domain

- In a Windows Domain, passwords hashes are used to authenticate users on hosts in the domain
- Password hashes are cached to avoid asking for the password
- Gives rise to devastating attack (Pass-the-Hash)
  - Obtain user credentials for one host in the domain (eg phishing)
  - Exploit vulnerability to become local administrator
  - Install process which waits for domain administrator to login into this machine
  - Extract cached hash for domain administrator
  - Login as domain administrator
- Defence mechanism exist but are painful to use
- `ssh` much better: public key on untrusted machine, private key on trusted machine

Getting Windows Password Hashes

- Boot into Linux
- Get SAM file

Password crackers

- John the Ripper
  - Most common brute force cracker
  - Open source
- Hashcat
  - Claims to be the fastest/best.
- Ophacrack
  - State of the art, free, rainbow table software.

Password capture by attacker

- **Phishing:** Username and password captured by attackers via malicious links (eg fake bank websites)
- used to login and then for attacks (ransomware, theft of credit card details, IP $\cdots$)
- Best protection: multi-factor authentication (something else apart from username and password, eg one-time password via apps, SMS codes, physical hardware tokens $\cdots$
- ssh with public key authentication only also protects against phishing

Password Injection

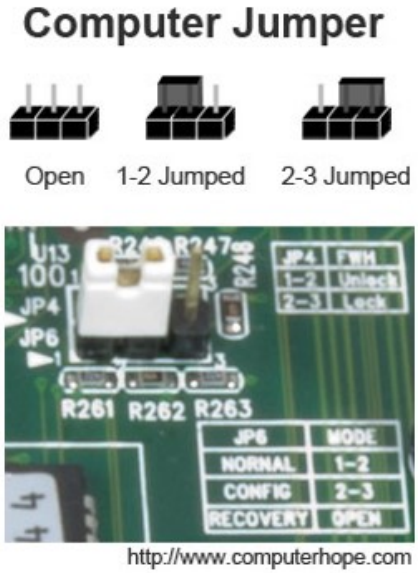
- Want access to the system without cracking the password?
- Have access to the hard disk?
- Add your own account, or replace the hash with one you know.

Better Security: BIOS

- Set a password in the BIOS to stop the computer booting from anything but the hard disk.
- It is very hard to brute force the BIOS.
- Workaround: remove the hard disk from the computer or reset BIOS password.

Resetting the BIOS password

- BIOS password can be reset by opening the box.



Best Security

- Encryption of important file.
- Whole disk encryption
  - Encrypt the whole hard drive
  - Key can be brute forced
  - Not safe if the computer is in sleep mode.
- E.g. BitLocker, FileVault, Luks

Summary

- Discussed mechanisms for access control, eg file permissions, mechanisms for granting higher privileges temporarily
- Authentication still mostly done with username and password
- However, subject to phishing attacks
  Multifactor authentication and public key ssh access protect against this
- Need also to protect hard disk against loss or theft
  Best way: use full disk encryption