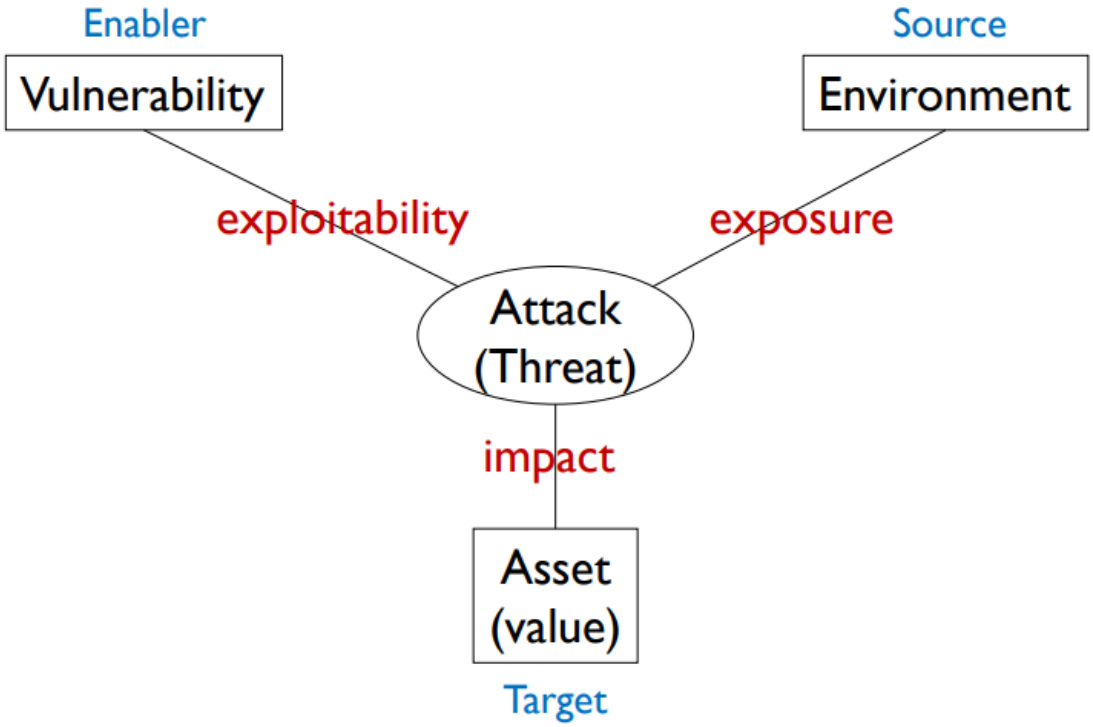
C2 Measuring Security

1. Risk Analysis

- Risk analysis enables identification of assets, vulnerabilities and threats in order to make informed decisions about which control to use
- There is a need to balance **cost of security** and **cost of insecurity**



2. Risk Analysis for Software Development

- **Threat Modelling**
  - Helps understand where the product is most vulnerable
  - Helps find design bugs that are not likely to be found when looking at individual components

2.1 Threat Modelling Process

1. Assemble Threat Modelling team
   - Regular threat modelling meetings with people from product group
   - Most security savvy person leads the team
   - One member from each team should be involved at meetings (design, coding, testing, documentation)
   - **do not fix problems during meetings**
2. Formally Decompose Application

    - Identify **key components**, **security boundaries**, **data flow** between components
    - **Data Flow Diagrams**
        - A system can be decomposed into subsystems which can be further decomposed into lower-level subsystems
        - Identify boundaries between trusted and untrusted components using a high level context diagram (usually contains only **one process and no data store**, modelling the users or external entities interacting with system)
3. Determine Threats
4. Rank Threats
5. Mitigate Threats
    - How to respond
    - Mitigation techniques
    - Technologies to implement techniques

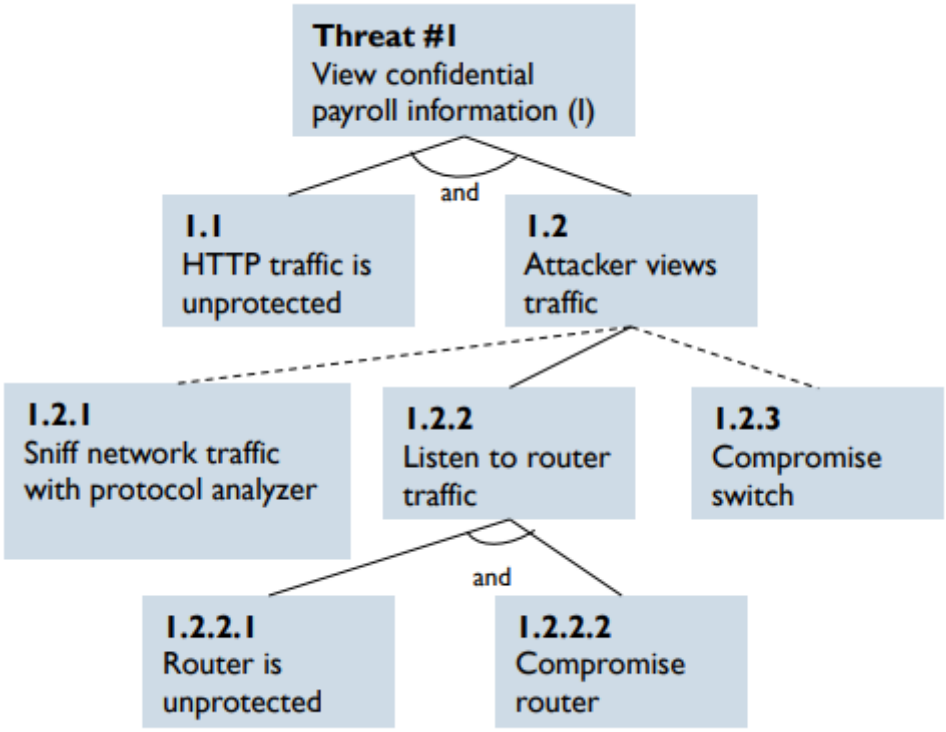
always ask if C.I.A. has been compromised

2.2 STRIDE

- Categorises threats by their effects, threats listed are interrelated
- **Spoofing Identity** (impersonation / masquerading)
    - Attacker poses as another entity
- **Tampering with data** (data integrity)
    - Unauthorised / Malicious modification of data
- **Repudiation** (accountability)
    - User (wrongly) denies having performed an action, and there is insufficient evidence to prove they did
- **Informal Disclosure** (confidentiality)
    - Information revealed to unauthorised entity
- **Denial of Service** (availability)
    - Deny service to **authorised** user
    - Can be anonymous
    - Rarely stopped completely, but can be made expensive for an attacker
- **Elevation of Privilege**
    - User gains more privileges than entitled, has sufficient access to compromise or destroy entire system

2.3 Threat Trees (Attack trees)

- Threat trees further organises threat analysis, describes the decision-making process of an attacker
- Threats are identified using STRIDE, threat trees follow up by determining how threats can manifest itself



- Dotted lines are used to show least likely attack points, solid lines are used for the most likely

Risk = Damage Potential * Likelihood

2.4 DREAD

- **Damage Potential**
  - How great can the damage be?
  - How valuable is the data that is affected?
- **Reproducibility**
  - How easy is it to get the attack to work in the wild?
- **Exploitability**
  - How much effort and expertise is required to mount an attack?
  - Can attack be automated
- **Affected Users**
  - How many users would be affected?
  - Does attack affect client or servers?
  - Does attack only work in special configurations?

- **Discoverability**
  - Will vulnerability be discovered by potential hackers?

2.5 Common Vulnerability Scoring System (CVSS)

- **Cumulative Voting**
  - Helps avoid giving high scores too all components of DREAD which was not helpful for prioritising risks
  - Fixed number of votes that can be assigned to identified risks, have to split votes up to prioritise different risks
- Starts from the vulnerabilities when organising impact assessment
- **Scoring Scheme**

Basic metrics		Temporal metrics	Environmental metrics	
Attack vector	Confidentiality impact	Exploit code maturity (exploitability)	Collateral damage potential	Confidentiality requirement
Attack complexity	Integrity impact	Remediation level	Target distribution	Integrity requirement
Privilege Required	Availability impact	Report confidence		Availability requirement

Starting from functional decomposition, each component's threats is determined using STRIDE, which will then be followed up by a Threat Tree to determine how the threats can manifest, then DREAD + CVSS is used to rank threats in terms of priority

2.6 Responding to Threats

- **Do Nothing**
  - May only delay the pain
  - If removing the problem would break an important application, doing nothing would be considered
- **Warn User**
  - Tell users they are about to do something dangerous and let them decide whether to go ahead
- **Remove Problem**
  - Remove dangerous feature from product
- **Fix Problem**
- Techniques are not the same as technologies
  - A technique is derived from a high-level appreciation of what kinds of technologies can be applied to mitigate a threat

Threat Type	Mitigation Techniques
Spoofing identity	Appropriate authentication, Protect secret data
Tampering with data	Appropriate authorization, Hashes, MACs, Digital signatures, Tamper-resistant protocols
Repudiation	Digital signatures, Timestamps, Audit trails
Information disclosure	Authorization, Privacy-enhanced protocols, Encryption, Protect secrets
Denial of service	Appropriate authentication, authorization, Filtering, Throttling, Quality of service
Elevation of privilege	Run with least privilege

3. Attack Surfaces (skipped in lecture)

- The sum of the different points (attack vectors) where an attacker can try to enter data / extract data from
- **3 Dimensions of attack surfaces include targets & enablers, channels & protocols, access rights**
- Attackability is the measure of how exposed a system's attack surface is
- Reducing attack surface = eliminating / reducing types or instances of **targets and attacks**

3.1 Computing Attack Surfaces

- Attack surface sum is the sum of independent weighted contributions from
  - set of channel types
  - set of process target types
  - set of data target types
  - set of process enablers
  - all subject to the constraints of the access rights relation
- Good for comparing against closely related systems