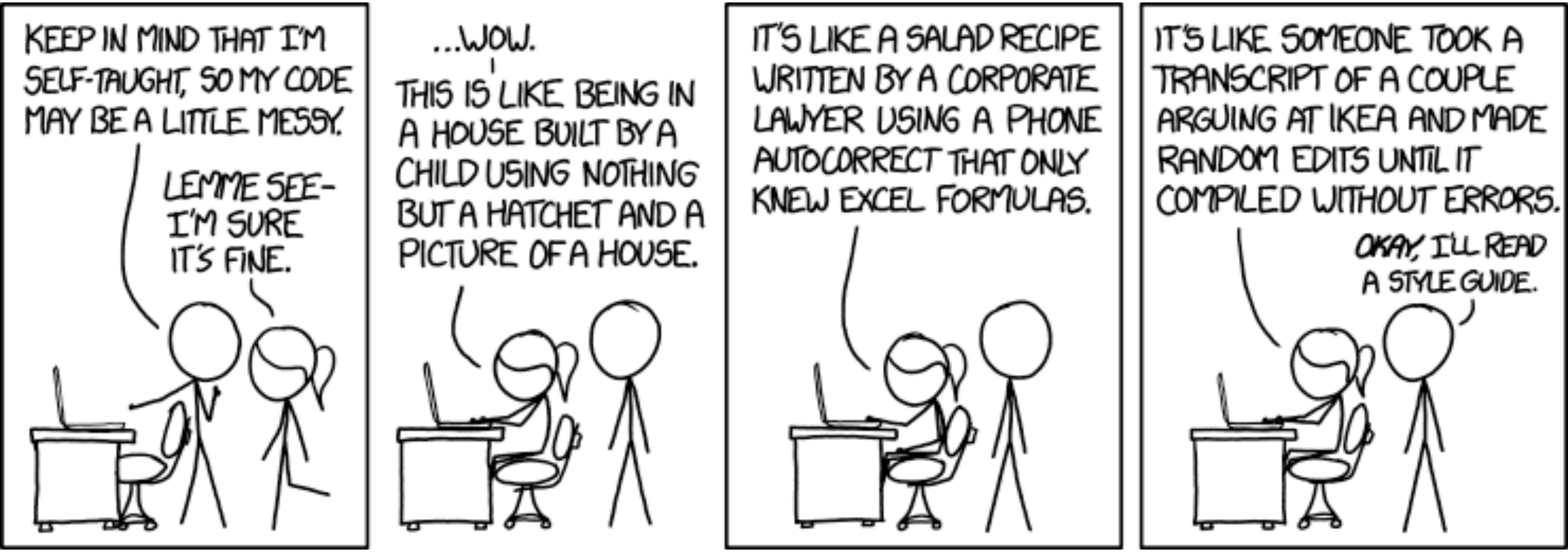
CS 55: Security and Privacy

Secure systems development


KEEP IN MIND THAT I'M SELF-TAUGHT, SO MY CODE MAY BE A LITTLE MESSY.
LEMME SEE- I'M SURE IT'S FINE.
...WOW.
THIS IS LIKE BEING IN A HOUSE BUILT BY A CHILD USING NOTHING BUT A HATCHET AND A PICTURE OF A HOUSE.
IT'S LIKE A SALAD RECIPE WRITTEN BY A CORPORATE LAWYER USING A PHONE AUTOCORRECT THAT ONLY KNEW EXCEL FORMULAS.
IT'S LIKE SOMEONE TOOK A TRANSCRIPT OF A COUPLE ARGUING AT IKEA AND MADE RANDOM EDITS UNTIL IT COMPILED WITHOUT ERRORS.
OKAY, I'LL READ A STYLE GUIDE.

https://xkcd.com/1513/

Agenda

1. Safety vs security
2. Risk management
3. Threat modeling
4. Best practices and common failures
5. Prioritization

First a little humor, but with some important messages embedded

Murphy's Laws

- Nothing is as easy as it looks.
- Everything takes longer than you think.
- **Anything that can go wrong will go wrong.**
- If there is a possibility of several things going wrong, the one that will cause the most damage will be the one to go wrong. Corollary: If there is a worse time for something to go wrong, it will happen then.
- If anything simply cannot go wrong, it will anyway.
- If you perceive that there are four possible ways in which a procedure can go wrong, and circumvent these, then a fifth way, unprepared for, will promptly develop.
- Left to themselves, things tend to go from bad to worse.
- If everything seems to be going well, you have obviously overlooked something.
- Nature always sides with the hidden flaw.
- It is impossible to make anything foolproof because fools are so ingenious.
- Whenever you set out to do something, something else must be done first.
- The Light at the end of the tunnel is only the light of an oncoming train.

https://www.cs.cmu.edu/~fgandon/miscellaneous/murphy/

First a little humor, but with some important messages embedded

Murphy's Laws

- Nothing is as easy as it looks.
- Everything takes longer than you think.
- Anything that can go wrong will go wrong.
- **If there is a possibility of several things going wrong, the one that will cause the most damage will be the one to go wrong. Corollary: If there is a worse time for something to go wrong, it will happen then.**
- If anything simply cannot go wrong, it will anyway.
- If you perceive that there are four possible ways in which a procedure can go wrong, and circumvent these, then a fifth way, unprepared for, will promptly develop.
- Left to themselves, things tend to go from bad to worse.
- If everything seems to be going well, you have obviously overlooked something.
- Nature always sides with the hidden flaw.
- It is impossible to make anything foolproof because fools are so ingenious.
- Whenever you set out to do something, something else must be done first.
- The Light at the end of the tunnel is only the light of an oncoming train.

https://www.cs.cmu.edu/~fgandon/miscellaneous/murphy/

First a little humor, but with some important messages embedded

Murphy’s Laws

- Nothing is as easy as it looks.
- Everything takes longer than you think.
- Anything that can go wrong will go wrong.
- If there is a possibility of several things going wrong, the one that will cause the most damage will be the one to go wrong. Corollary: If there is a worse time for something to go wrong, it will happen then.
- **If anything simply cannot go wrong, it will anyway.**
- If you perceive that there are four possible ways in which a procedure can go wrong, and circumvent these, then a fifth way, unprepared for, will promptly develop.
- Left to themselves, things tend to go from bad to worse.
- If everything seems to be going well, you have obviously overlooked something.
- Nature always sides with the hidden flaw.
- It is impossible to make anything foolproof because fools are so ingenious.
- Whenever you set out to do something, something else must be done first.
- The Light at the end of the tunnel is only the light of an oncoming train.

https://www.cs.cmu.edu/~fgandon/miscellaneous/murphy/

First a little humor, but with some important messages embedded

Murphy's Laws

- Nothing is as easy as it looks.
- Everything takes longer than you think.
- Anything that can go wrong will go wrong.
- If there is a possibility of several things going wrong, the one that will cause the most damage will be the one to go wrong. Corollary: If there is a worse time for something to go wrong, it will happen then.
- If anything simply cannot go wrong, it will anyway.
- **If you perceive that there are four possible ways in which a procedure can go wrong, and circumvent these, then a fifth way, unprepared for, will promptly develop.**
- Left to themselves, things tend to go from bad to worse.
- If everything seems to be going well, you have obviously overlooked something.
- Nature always sides with the hidden flaw.
- It is impossible to make anything foolproof because fools are so ingenious.
- Whenever you set out to do something, something else must be done first.
- The Light at the end of the tunnel is only the light of an oncoming train.

https://www.cs.cmu.edu/~fgandon/miscellaneous/murphy/

First a little humor, but with some important messages embedded

Murphy's Laws

- Nothing is as easy as it looks.
- Everything takes longer than you think.
- Anything that can go wrong will go wrong.
- If there is a possibility of several things going wrong, the one that will cause the most damage will be the one to go wrong. Corollary: If there is a worse time for something to go wrong, it will happen then.
- If anything simply cannot go wrong, it will anyway.
- If you perceive that there are four possible ways in which a procedure can go wrong, and circumvent these, then a fifth way, unprepared for, will promptly develop.
- Left to themselves, things tend to go from bad to worse.
- If everything seems to be going well, you have obviously overlooked something.
- Nature always sides with the hidden flaw.
- **It is impossible to make anything foolproof because fools are so ingenious.**
- Whenever you set out to do something, something else must be done first.
- The Light at the end of the tunnel is only the light of an oncoming train.

https://www.cs.cmu.edu/~fgandon/miscellaneous/murphy/

Discussion

What is the difference between safety and security?

What are some examples of each?

What do we protect?

Why can companies and governments spend billions of dollars and still be vulnerable to risks?

Safety focuses on protection from harm not caused by malicious intent

https://www.youtube.com/watch?v=3LiGsPR34w8&start=134

Agenda

1. Safety vs security

➡ 2. Risk management

3. Threat modeling

4. Best practices and common failures

5. Prioritization

Critical systems must be protected from both safety and security issues

Critical systems

- Safety-critical
- Business-critical
- Security-critical

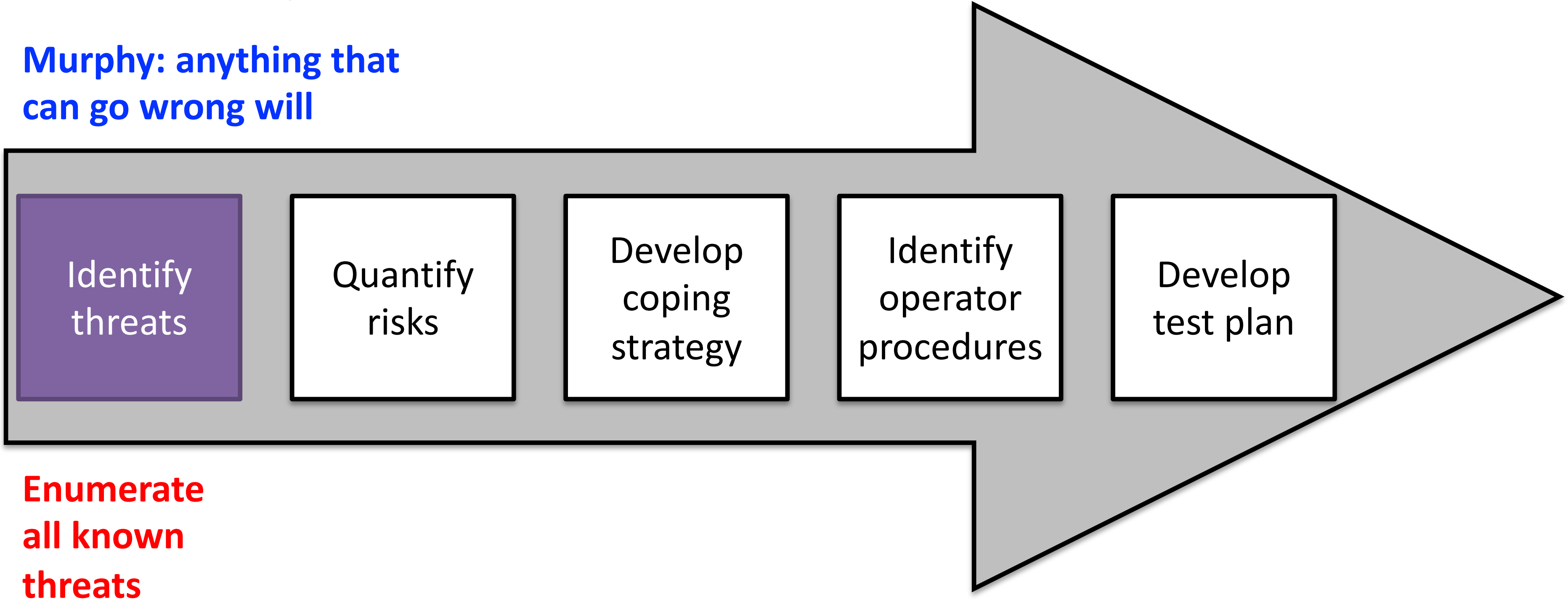
Adapted from Anderson, Ross. *Security engineering: a guide to building dependable distributed systems*. John Wiley & Sons, 2020.

Risk management is a process that attempts to minimize the negative impacts

Critical systems

- Safety-critical
- Business-critical
- Security-critical

Murphy: anything that can go wrong will



Enumerate all known threats

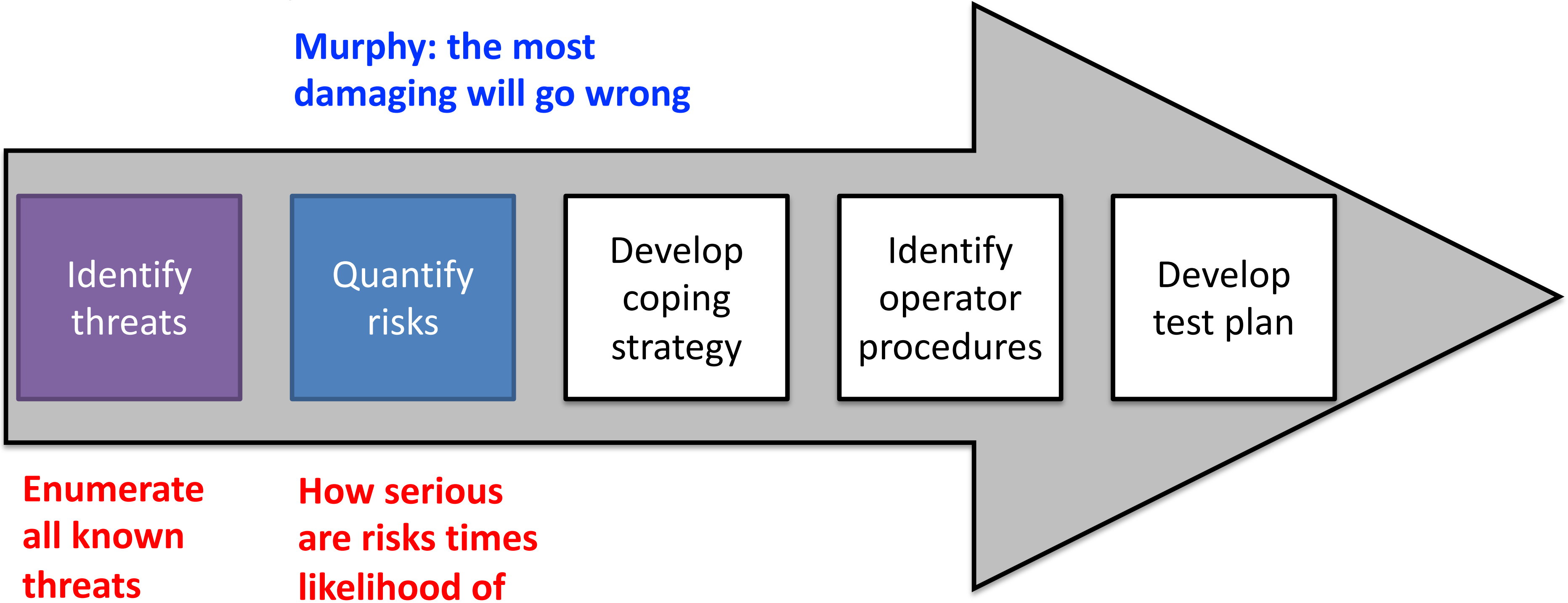
Adapted from ISO 61508 and Anderson, Ross. *Security engineering: a guide to building dependable distributed systems*. John Wiley & Sons, 2020.

Risk management is a process that attempts to minimize the negative impacts

Critical systems

- Safety-critical
- Business-critical
- Security-critical

Murphy: the most damaging will go wrong



Enumerate all known threats

How serious are risks times likelihood of occurring

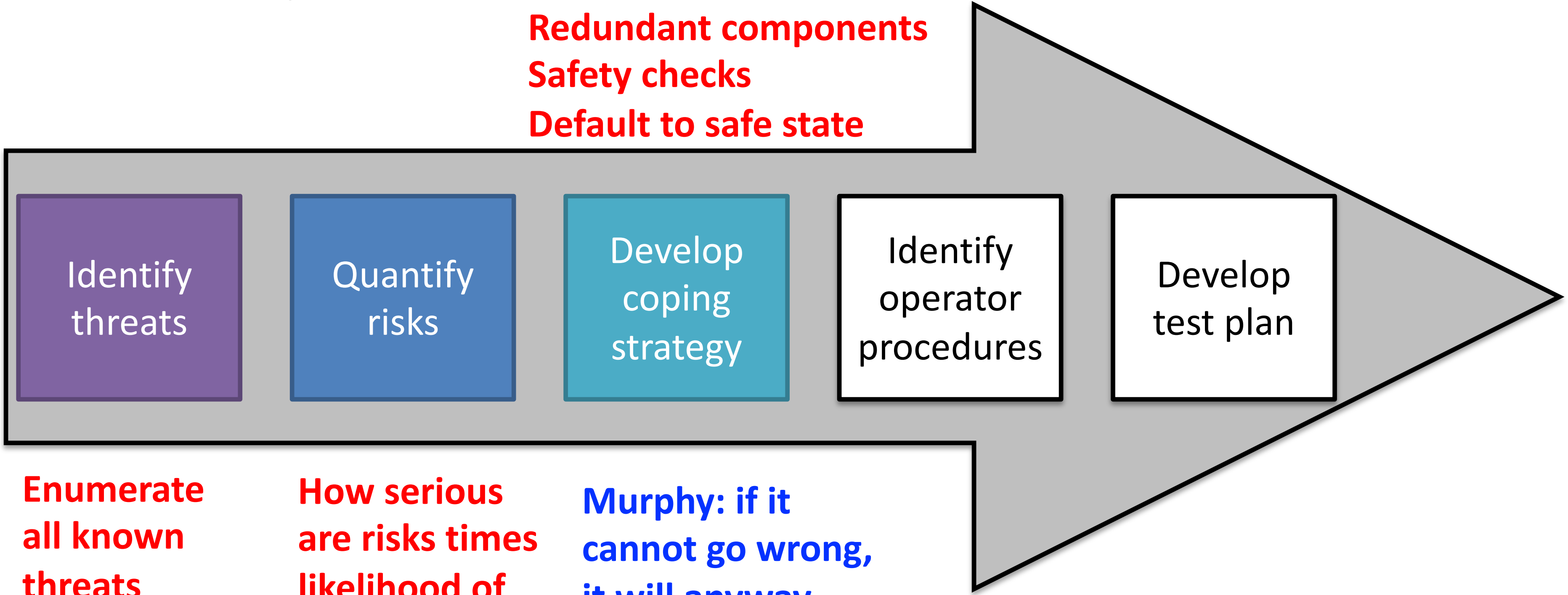
Adapted from Anderson, Ross. *Security engineering: a guide to building dependable distributed systems*. John Wiley & Sons, 2020.

Risk management is a process that attempts to minimize the negative impacts

Critical systems

- Safety-critical
- Business-critical
- Security-critical

Redundant components
Safety checks
Default to safe state



Identify threats

Quantify risks

Develop coping strategy

Identify operator procedures

Develop test plan

Enumerate all known threats

How serious are risks times likelihood of occurring

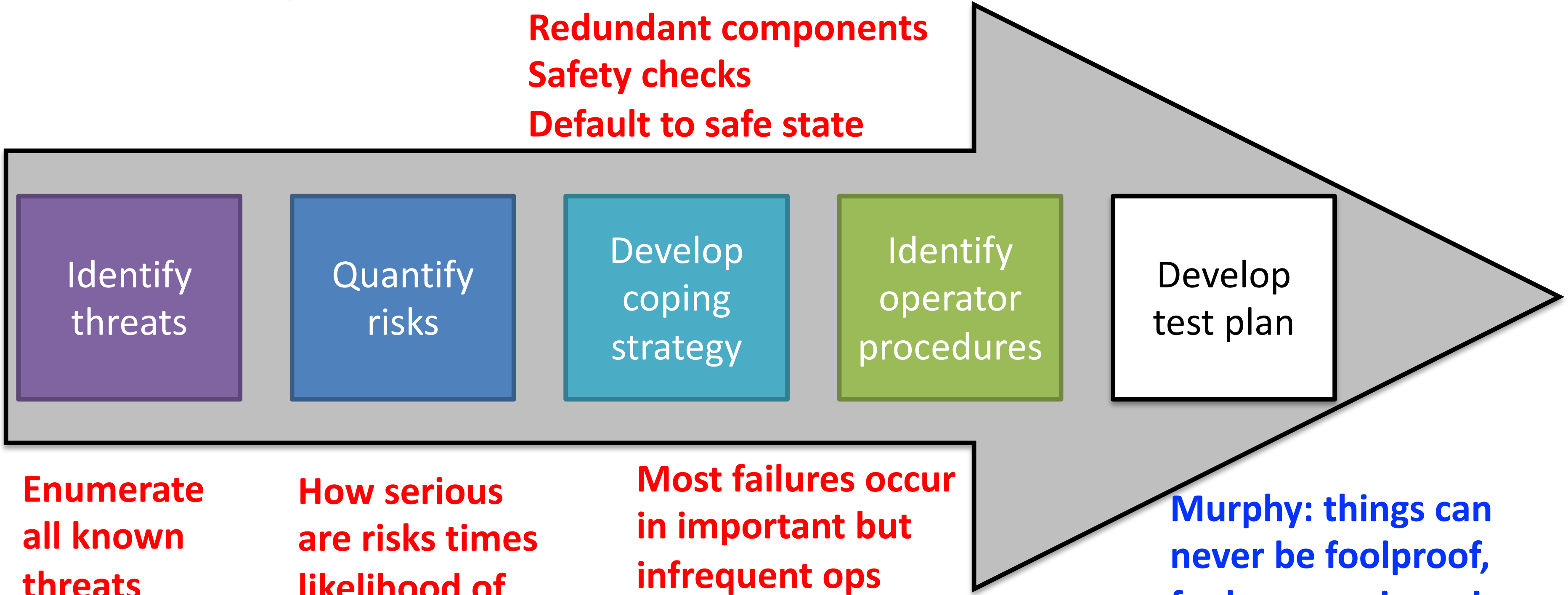
Murphy: if it cannot go wrong, it will anyway

Adapted from Anderson, Ross. *Security engineering: a guide to building dependable distributed systems*. John Wiley & Sons, 2020.

Risk management is a process that attempts to minimize the negative impacts

Critical systems

- Safety-critical
- Business-critical
- Security-critical



Redundant components
Safety checks
Default to safe state

Enumerate all known threats

How serious are risks times likelihood of occurring

Most failures occur in important but infrequent ops
Notes, Warnings, Cautions

Murphy: things can never be foolproof, fools are so ingenious

Adapted from Anderson, Ross. *Security engineering: a guide to building dependable distributed systems*. John Wiley & Sons, 2020.

Risk management is a process that attempts to minimize the negative impacts

Critical systems

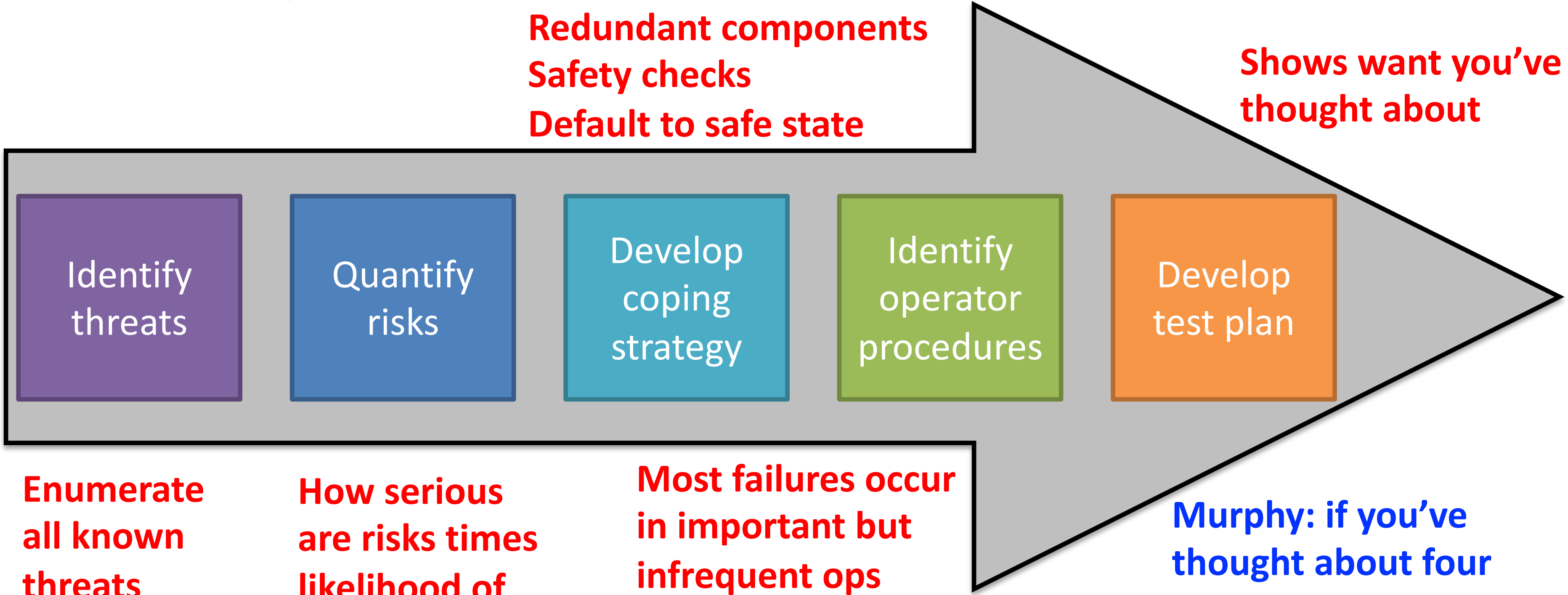
- Safety-critical
- Business-critical
- Security-critical

A safety engineer is concerned about MTBF of 10^9 hours

A security engineer worries whether an adversary can force the preconditions for that one-in-a-billion failure

Redundant components
Safety checks
Default to safe state

Shows want you've thought about



Identify threats

Quantify risks

Develop coping strategy

Identify operator procedures

Develop test plan

Enumerate all known threats

How serious are risks times likelihood of occurring

Most failures occur in important but infrequent ops
Notes, Warnings, Cautions

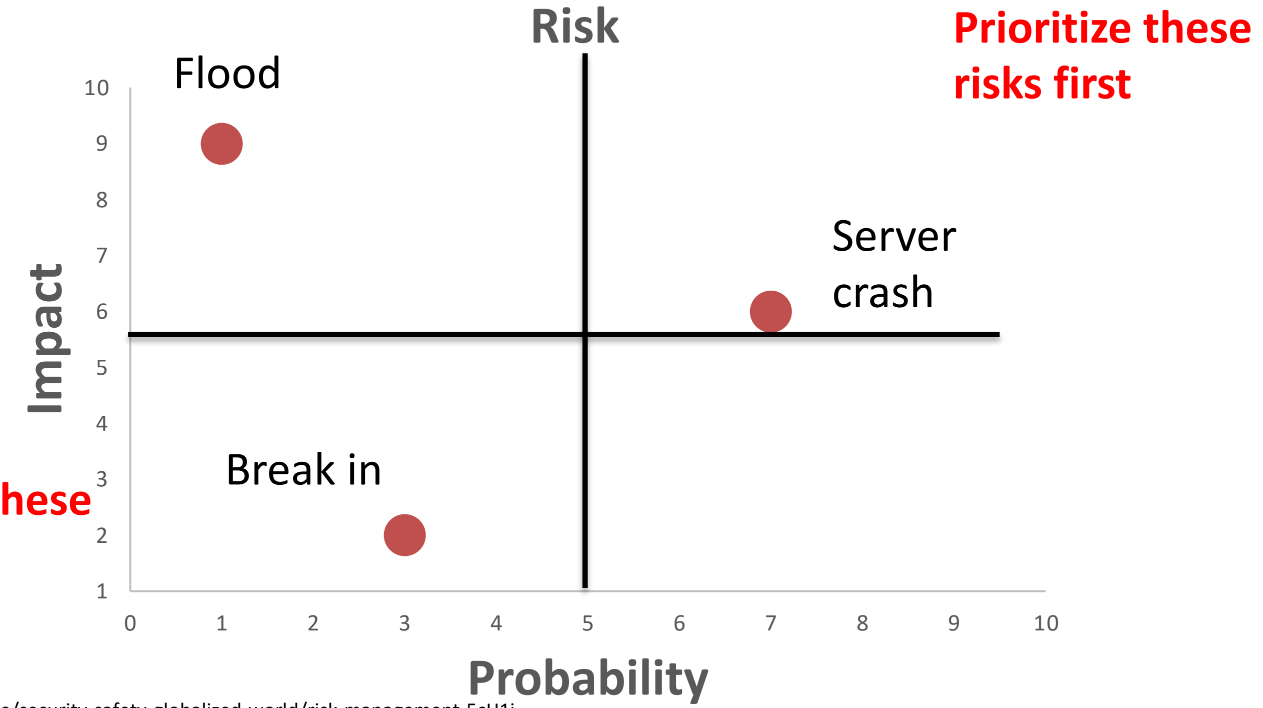
Murphy: if you've thought about four things that can go wrong, a fifth will appear

Adapted from Anderson, Ross. *Security engineering: a guide to building dependable distributed systems*. John Wiley & Sons, 2020.

Risk management views risk as something that can be managed

Can we eliminate all risks in real-world systems?

- Idea: reduce risks to manageable level by spending money/time on worst risks first
- Problem: how to quantify risks?
- Risk = probability x impact



Prioritize these risks first

Prioritize these risks last

Adapted from https://www.coursera.org/lecture/security-safety-globalized-world/risk-management-5cH1i

Risk management has been criticized on several fronts

Risk identification

- Cannot know all potential risks
- Unknown unknown (cannot imagine some risk scenarios)
- False sense of confidence of our degree of control

Quantification gives illusion of science

- Numbers of assigned to probability and impact
- Change assumptions, get different numbers
- What do you have to believe
- Cannot quantify properly

Discounting

- Alternative scenarios sometimes discounted
- False sense of security

Adapted from https://www.coursera.org/lecture/security-safety-globalized-world/risk-management-critiques-and-alternatives-MILQo

Agenda

1. Safety vs security
2. Risk management
3. Threat modeling
4. Best practices and common failures
5. Prioritization

Figuring out how to start thinking about threats can feel like being lost in the woods

There are a huge number of potential threats

- Hackers
- Nation states
- Insiders
- Nature

There are dozens of frameworks

Each case is unique and there is no solid consensus on how proceed

Surprisingly, there is no agreed upon standard for a ‘threat model’

Threat modeling is about understanding ***causes*** of possible security issues and ***options*** for protecting a system

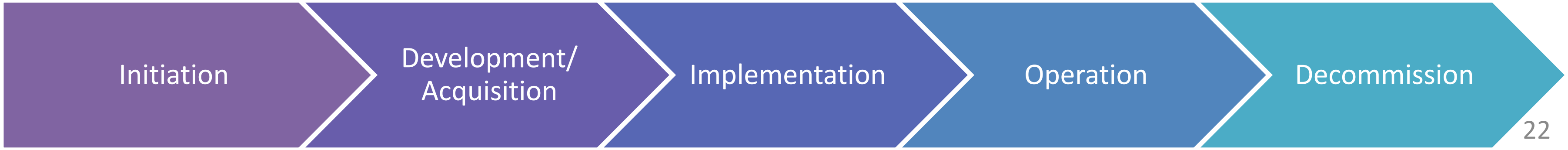
Ask three key questions about each module in a system

	Activity	**Outcome**
1) What are you building?	Explain and explore	Technical diagram
2) What can go wrong?	Brainstorm threats	A list of technical threats
3) What are you going to do?	Prioritize and fix	Prioritized fixes added to list of projects

Involve members from technical and business roles

Model parts of the system at first (don't try to model whole system at first)

Start when considering a new system and continue through lifespan

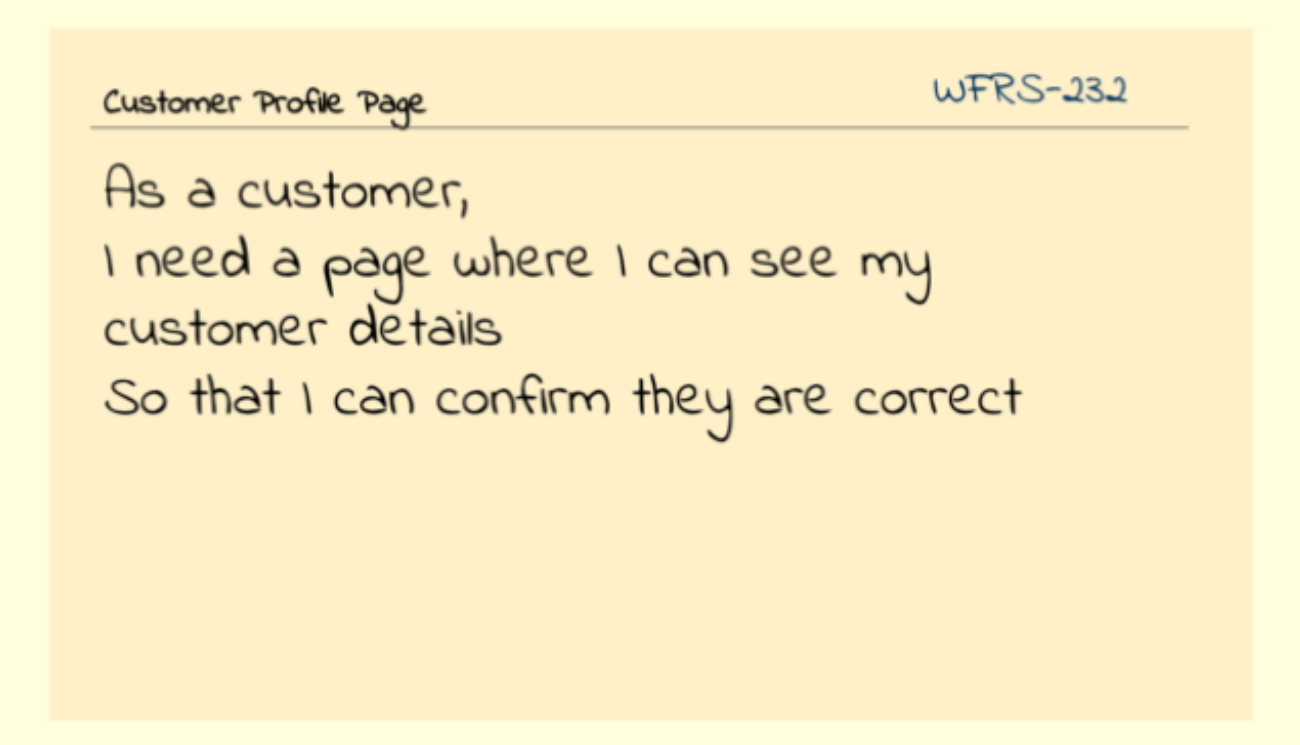


Adpated from https://martinfowler.com/articles/agile-threat-modelling.html and NIST SP 800-160

Example: consider a customer profile page

Development team at retail organization is building a platform to sell groceries

This is the epic they have in an upcoming sprint

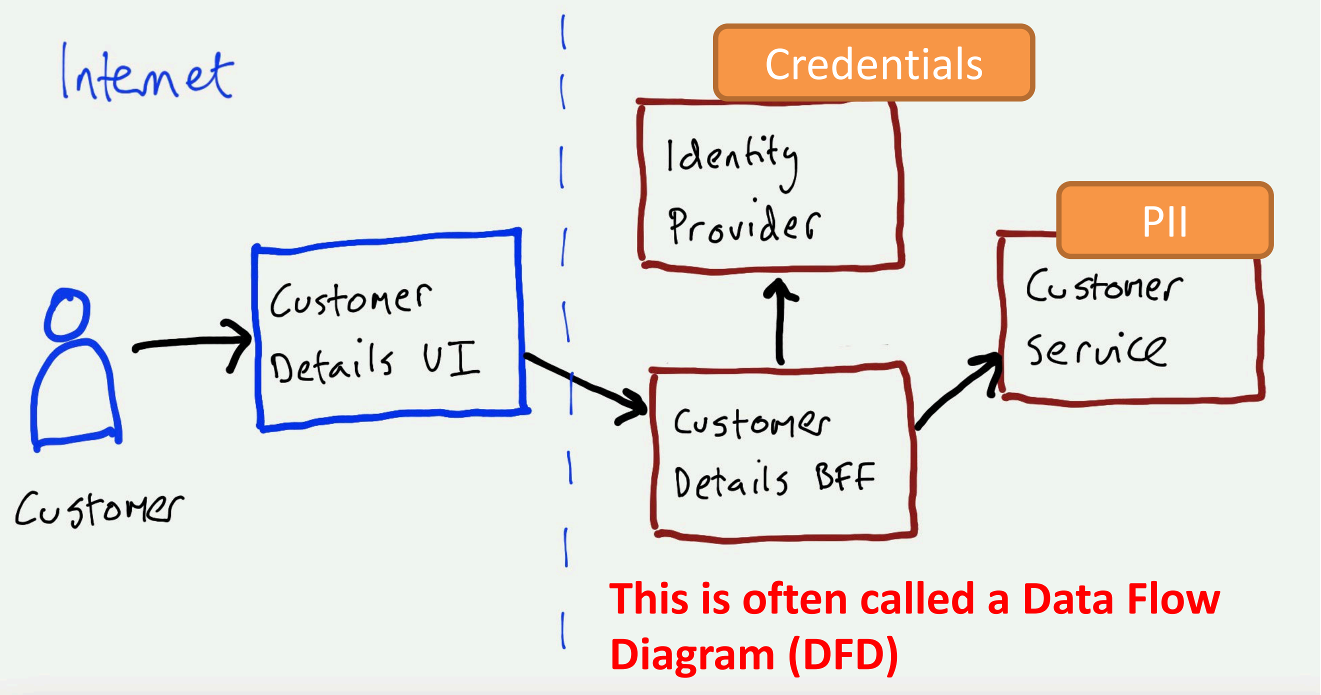


Customer Profile Page WFRS-232

As a customer,
I need a page where I can see my customer details
So that I can confirm they are correct

Adpated from https://martinfowler.com/articles/agile-threat-modelling.html and NIST SP 800-160

Question 1: What are you building?



This is often called a Data Flow Diagram (DFD)

Draw a diagram of all components

- Show data flow
- Label inside and outside network components
- Microsoft has a tool for this diagraming exercise
- Show diagram to non-tech business users

Data flows indicate pathways an attacker could abuse

Arrows indicate where request begins (but are often bi-directional)

Add assets to highlight what information must be protected

Adpated from https://martinfowler.com/articles/agile-threat-modelling.html and NIST SP 800-160

Question 2: What can go wrong?

STRIDE	Question	Concept involved
Spoofed identity	How can an adversary appear as a legitimate user (e.g., packet spoofing)?	• Identity • Authentication
Tampering with input	How can user input be abused to change system (e.g., buffer overflow, SQL injection, XSS)?	• Integrity • Availability
Repudiation of action	How can a user deny action?	• Integrity
Information disclosure	How can a user see information they shouldn't? (e.g., sniffing)	• Confidentiality
Denial of service	How can a user make the system unavailable?	• Availability
Elevation of privilege	How can a user obtain greater system rights?	• Authorization • Confidentiality • Integrity • Availability

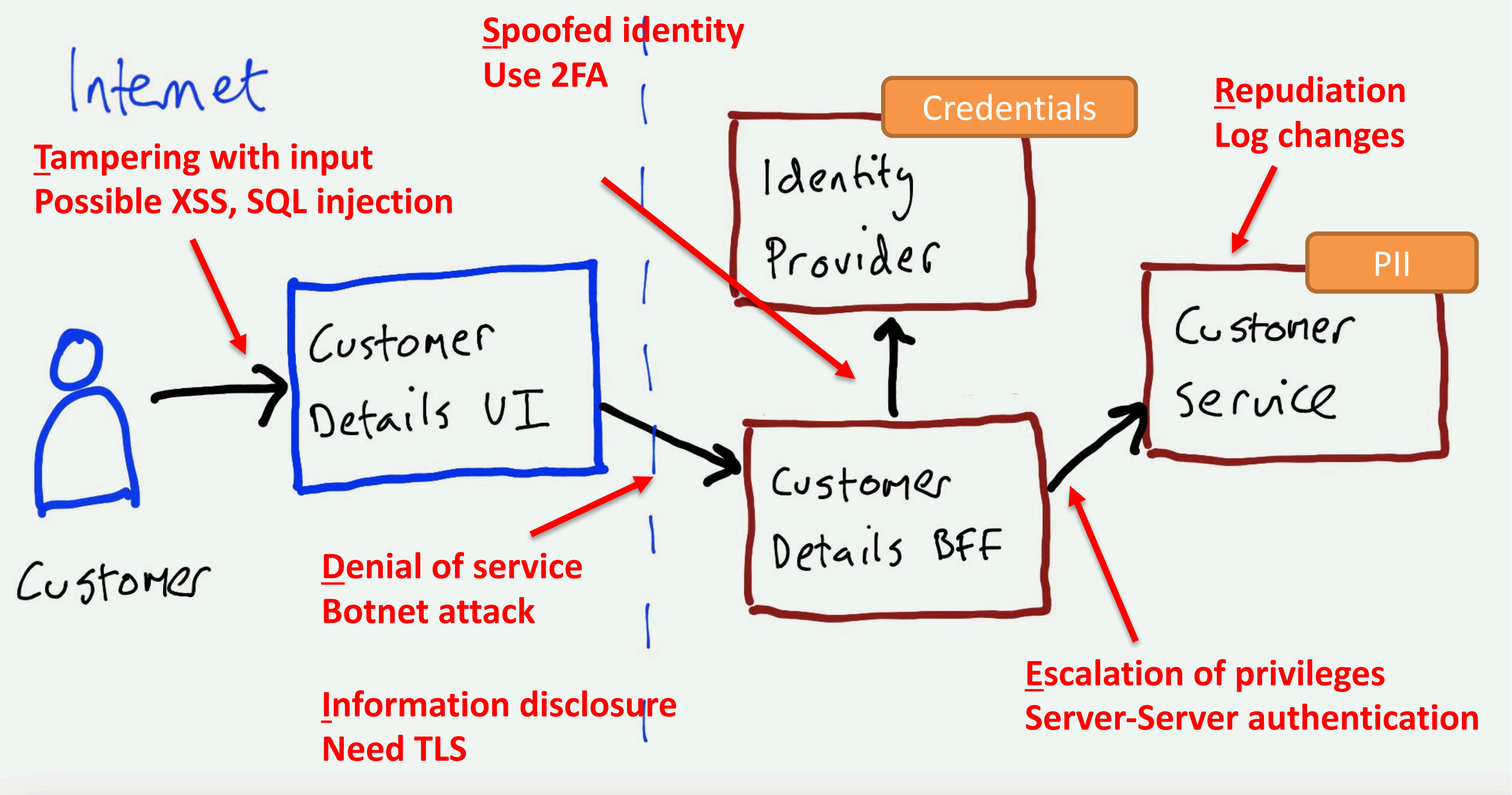
There are many other frameworks other than STRIDE (see https://insights.sei.cmu.edu/sei_blog/2018/12/threat-modeling-12-available-methods.html)

Brainstorm ways components could be compromised

Use the data flows line to consider interactions between components

Adpated from https://martinfowler.com/articles/agile-threat-modelling.html and NIST SP 800-160

Question 2: What can go wrong?



Spoofed identity
Use 2FA

Tampering with input
Possible XSS, SQL injection

Repudiation
Log changes

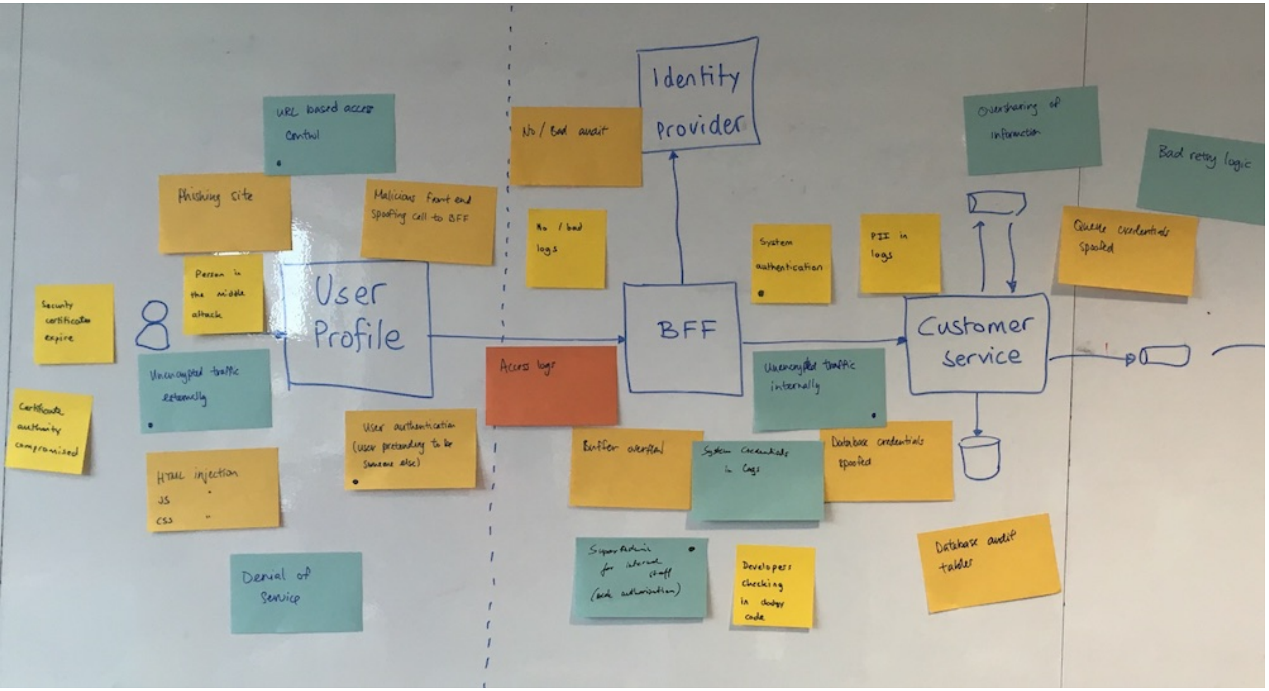
Denial of service
Botnet attack

Information disclosure
Need TLS

Escalation of privileges
Server-Server authentication

Adpated from https://martinfowler.com/articles/agile-threat-modelling.html and NIST SP 800-160

Question 3: What are you going to do about it?


URL based access Control
Phishing site
Malicious front end spoofing call to BFF
No / Bad audit
Identity Provider
No / bad logs
Person in the middle attack
Security certificates expire
User Profile
Unencrypted traffic externally
Certificate authority compromised
HTML injection
JS
CSS
User authentication (user pretending to be someone else)
Access logs
BFF
Buffer overflow
Denial of Service
Super Admin for internal staff (weak authorisation)
System Credentials in Logs
Developers checking in dodgy code
System authentication
PII in logs
Unencrypted traffic internally
Database credentials spoofed
Oversharing of information
Customer Service
Queue credentials spoofed
Bad retry logic
Database audit tables

Need to prioritize what to fix

- Assess business value of compromise
- Consider what are the main threats (fraud, malicious insiders, hackers)

- Vote for top riskiest threats
- Identify riskiest threats and countermeasure fixes
- Add fixes to work queue

Adpated from https://martinfowler.com/articles/agile-threat-modelling.html and NIST SP 800-160

DREAD can help you prioritize what to fix

DREAD	Rough scale
Damage potential	5 = information disclosures valuable with other vulnerabilities 8 = individual non-sensitive user data is compromised 9 = Administrative non-sensitive data is compromised 10 = Complete system or data destruction 10 = Application unavailable
Reproducibility	0 = Very hard or impossible, even for admins 5 = Complex steps required for authorized user 7.5 = Easy steps for authenticated user 10 = Anyone can do it
Exploitability	2.5 = Advanced programming/networking knowledge/tools 5 = Exploit exists in public, using available attack tools 10 = Available and no special tools required
Affected users	3 = Only one individual is compromised 8 = some users, but not all 9 = Admin users 10 = All users
Discoverability	0 = Very hard, requires source code or admin access 5 = Can figure it out by monitoring and manipulating requests 8 = Details already in public domain, can be easily discovered 10 = Information is visible from web browser

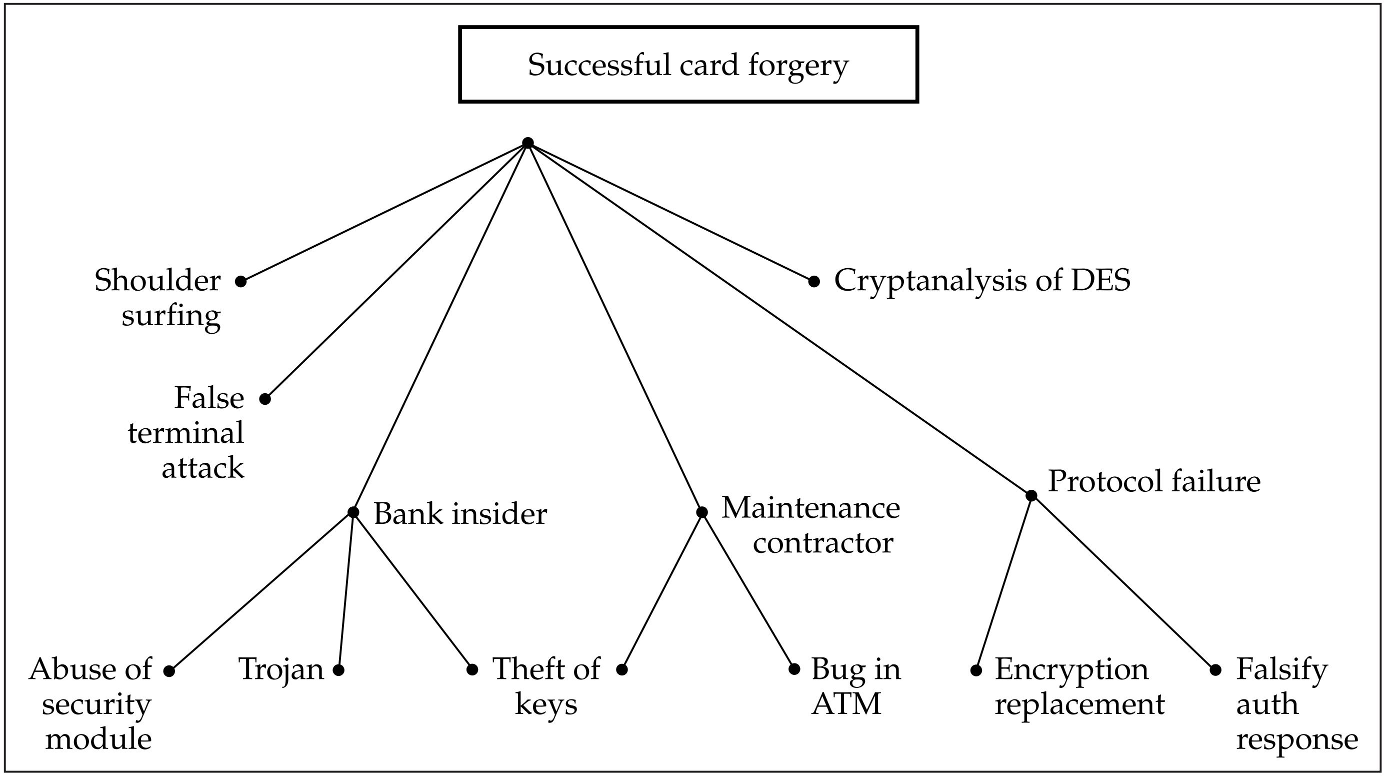
DREAD

- Score each factor on 0-10 scale
- Add scores together
- Divide by 5
- Fix most DREADful issues first

Some people drop the last D (say it favors security through obscurity)

https://haiderm.com/application-threat-modeling-using-dread-and-stride/

Fault/threat trees are another way to identify points of failure or compromise



Useful when there a small number of undesirable outcomes possible

Develop a threat tree for each undesirable outcome

Identify all causes of fault/failure

Work backward to find causes of the causes…

Anderson, Ross. *Security engineering: a guide to building dependable distributed systems*. John Wiley & Sons, 2020.

Agenda

1. Safety vs security
2. Risk management
3. Threat modeling
4. Best practices and common failures
5. Prioritization

Following best practices can reduce risk

1. **Least privilege**
   - Each user and each program should operate with the fewest/lowest privileges possible
   - Damage minimized in malicious or inadvertent compromise
2. **Economy of mechanism**
   - The design of the protection system should be small, simple, and straightforward
   - Such systems can be analyzed, tested, and verified
3. **Open design**
   - No security through obscurity (assume the adversary knows how the system works)
   - Make design available for public scrutiny

Adapted from *Security in Computing* 5th edition, by Pfleeger, Pfleeger, and Margulies

Following best practices can reduce risk

4. **Complete mediation**
   - Every access attempt must be checked
   - Checks cannot be bypassed

These ideas were developed in the '70s

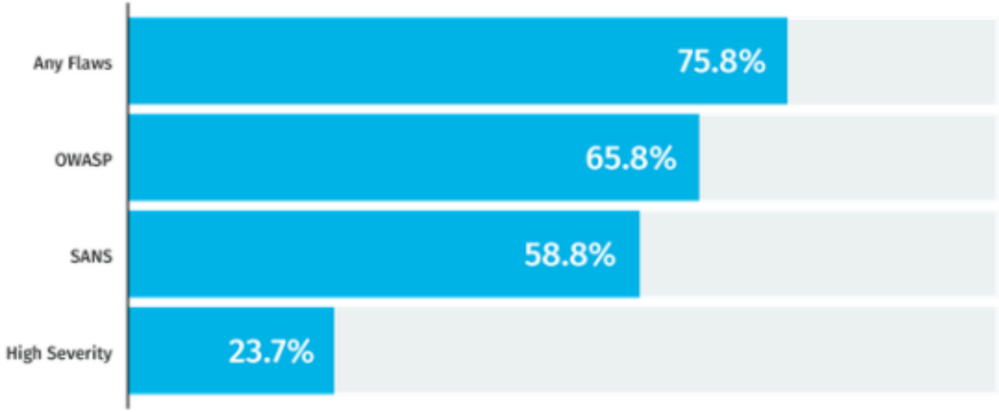
NIST now recommends 33 principles instead of these, see NIST SP 800-27

5. **Permission based**
   - The default condition should be denial of access
   - Identify what can be accessed, not what cannot be accessed
6. **Separation of privilege**
   - Access should depend on more than one condition (e.g., user authentication using two factors)
   - If one condition compromised, do not get access
7. **Least common mechanism**
   - Limit shared objects, use physical/logical separation
8. **Ease of use**
   - People will avoid/subvert mechanisms that are too hard to use

OWASP identifies the Top 10 web application security mistakes

The Open Web Application Security Project® (OWASP)

- Nonprofit foundation that works to improve the security of software
- Identifies most Top 10 common security flaws in deployed systems



Scan of 130,000 applications found over 65% had a security flaw in OWASP’s Top 10

Adapted from https://owasp.org/www-project-top-ten/ and https://www.veracode.com/security/owasp-top-10

OWASP identifies the Top 10 web application security mistakes

1. **Injection**
   - When untrusted data is used as part of a command or query
   - CS55: SQL injection, buffer overflows
2. **Broken authentication**
   - Authentication and session management implemented incorrectly allow adversaries to compromise passwords, keys, …
   - CS55: authentication
3. **Sensitive data exposure**
   - Many applications do not properly protect financial, healthcare, PII data
   - CS55: multilevel security
4. **XML External Entities (XXE)**
   - Older or poorly configured XML processors use external entity references
5. **Broken access control**
   - Restrictions on what authenticated users can do are not properly enforced
   - CS55: authorization

Adapted from https://owasp.org/www-project-top-ten/

OWASP identifies the Top 10 web application security mistakes

6. **Security misconfiguration**
   - Most common issue, insecure default configs, or incomplete/ad hoc
   - CS55: penetration testing
7. **Cross site scripting**
   - When an application trusts data in a web page without validation
   - CS55: XSS
8. **Insecure deserialization**
   - Taking data structured from some format, and rebuilding it into an object
9. **Using components with known vulnerabilities**
   - Components such as libraries, frameworks, etc., run with app privileges
   - CS55: network scanning
10. **Insufficient logging and monitoring**
    - Allows adversaries to attack systems without effective incident response
    - CS55: IDS/IPS; P > D + R

Adapted from https://owasp.org/www-project-top-ten/

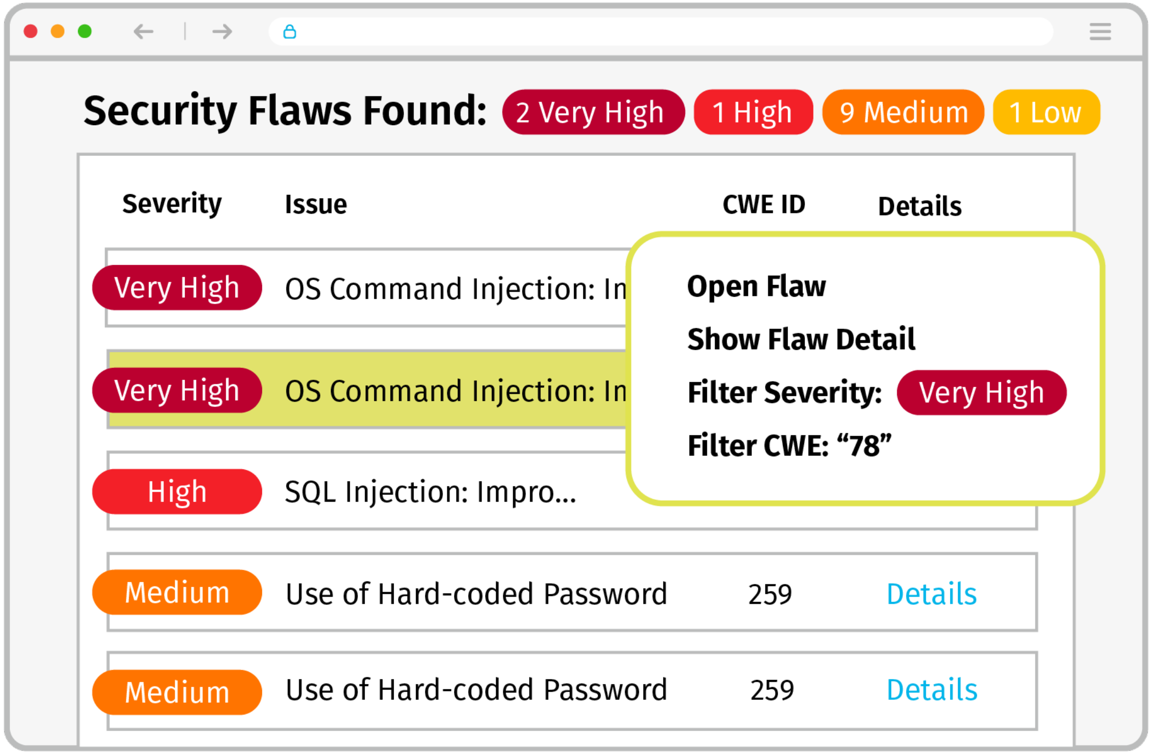
Static analysis tools can help detect some of these mistakes

Static Application Security Testing (SAST)

- Analyze source code
- Look for vulnerabilities
  - Buffer overflows
  - SQL injections
  - Others

Do not rely on this alone

- Example: cannot find poorly implemented crypto
- Use as defense in depth!



Security Flaws Found: 2 Very High 1 High 9 Medium 1 Low

Severity	Issue	CWE ID	Details
Very High	OS Command Injection: In		
Very High	OS Command Injection: In		
High	SQL Injection: Impro...		
Medium	Use of Hard-coded Password	259	Details
Medium	Use of Hard-coded Password	259	Details

Adapted from https://www.youtube.com/watch?v=pQdhsEVoHDA and https://www.veracode.com/products/binary-static-analysis-sast

Dynamic analysis tools can help find other mistakes

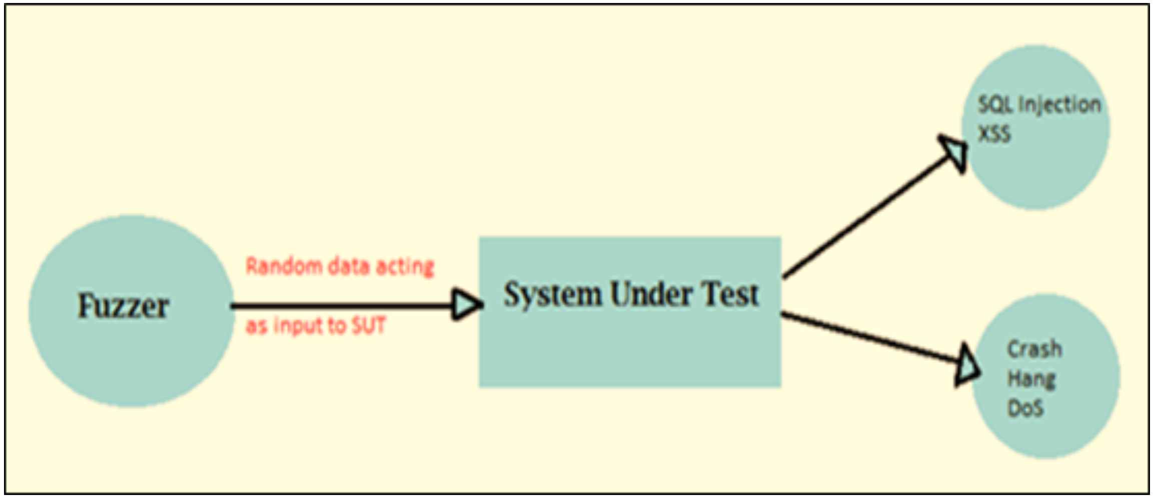
Dynamic analysis (fuzzing)

- Input data into app
- Look for unexpected/ unusual responses

Takes a lot of processor time

Stress testing

- Simulates many simultaneous users
- Make sure app can handle the load



Adapted from https://www.youtube.com/watch?v=pQdhsEVoHDA and https://www.veracode.com/products/binary-static-analysis-sast

Agenda

1. Safety vs security
2. Risk management
3. Threat modeling
4. Best practices and common failures
5. ➡ Prioritization

Sometimes it is difficult to prioritize security projects over other projects

VS

VERIFIED Non-GMO & Gluten Free
Centrum
SILVER
220 Tablets
MULTIVITAMIN/ MULTIMINERAL SUPPLEMENT
Adults 50+

Sometimes security projects are like vitamins (unless urgent threat occurs) – they pay off in the long run

Aspirin is easier to justify spending now to relief immediate pain

Example: should a retail store stop theft or increase sales?

Retail store

	Current State
Sales ($)	$ 10,000,000
Gross margin (%)	10%
Gross margin ($)	$ 1,000,000
Theft (%)	1.5%
Theft ($)	$ 150,000
Operating profit ($)	$ 850,000
Profit increase (%)	

Example: should a retail store stop theft or increase sales?

Retail store

	Current State	Stop theft
Sales ($)	$ 10,000,000	$ 10,000,000
Gross margin (%)	10%	10%
Gross margin ($)	$ 1,000,000	$ 1,000,000
Theft (%)	1.5%	**0.00%**
Theft ($)	$ 150,000	$ -
Operating profit ($)	$ 850,000	$ 1,000,000
Profit increase (%)		17.65%

Example: should a retail store stop theft or increase sales?

Retail store

	Current State	**Stop theft**	**Increase sales**
Sales ($)	$ 10,000,000	$ 10,000,000	**$ 11,764,706**
Gross margin (%)	10%	10%	10%
Gross margin ($)	$ 1,000,000	$ 1,000,000	$ 1,176,471
Theft (%)	1.5%	0.00%	1.5%
Theft ($)	$ 150,000	$ -	$ 176,471
Operating profit ($)	$ 850,000	$ 1,000,000	$ 1,000,000
Profit increase (%)		17.65%	17.65%

Company might be indifferent between stopping theft or increasing sales

If sales increases by a larger number, might prefer that approach even if theft increases

Will likely look at cost of each to calculate ROI

Pierson’s method of prioritizing projects

Every department in an organization has projects they consider vitally important

There are normally more projects in an organization that a technology department can simultaneously work on at any given time

Result: some projects wait, project owners grumble and fume

Department heads whose projects are not being addressed get angry… at you!

Pierson’s method of prioritizing projects

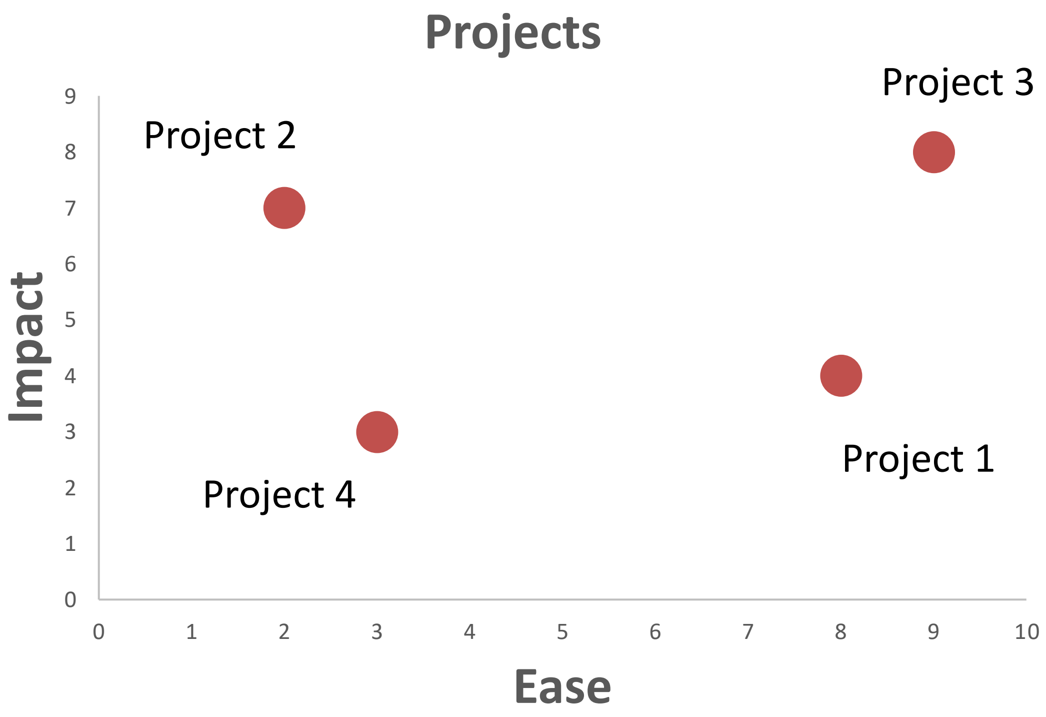
Quarterly meeting with all department heads

Each brings their most important projects

I rate each project on ease

The *__group__* rates each project on business impact

Plot projects on XY scatter graph



Note: business leaders decide *__together__* the importance to the overall *__organization__* of each project

Urgency baked into impact

Pro tip: do not put numbers on axis while discussing

Pierson’s method of prioritizing projects

Quarterly meeting with all department heads

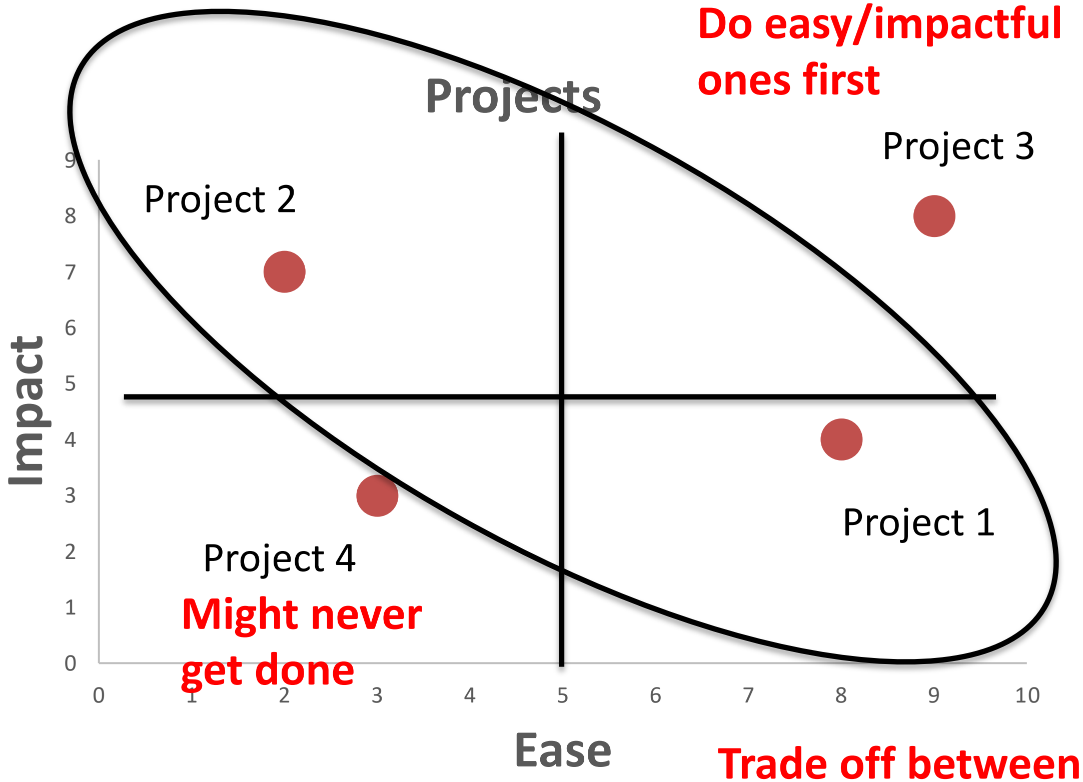
Each brings their most important projects

I rate each project on ease

The *group* rates each project on business impact

Plot projects on XY scatter graph

Repeat next quarter to ensure priorities haven’t changed



Do easy/impactful ones first

Might never get done

Break projects into quadrants

Trade off between easy/not impactful and hard/impactful

Now business leaders have a justification for how projects were prioritized