CSE127, Computer Security

Threat Modeling and Risk

UC San Diego

Housekeeping

General course things to know

- Due by **1/15** at 11:59
  - PA1! Get started early, so you can ensure the infrastructure you need for the PA is working properly.
- Due **1/16** at 11:59
  - #FinAid Canvas quiz: https://canvas.ucsd.edu/courses/71475/quizzes/238979, reminder to do this!
- Course staff office hours is now available on the website! Lots of OH throughout the week.
- Updated the website with recommended additional readings for the new few weeks

Previously on CSE127…

- We talked about **trust**: to have *security,* we must trust something (and for complete *security*, we must trust *everything*)
  - But it can be hard to trust **anything**, ranging from software to videos to news
- Question: **How do we reason about security in such a fractured trust ecosystem?**

Today's lecture – Security fundamentals, threat models, risk

Learning Objectives

- Understand what a threat model is, why we have threat models, and get some hands on experience with threat modeling
  - Get experience with the adversarial mindset
  - Evaluate potential mitigation options
  - Analyze tradeoffs
- Understand CIA — confidentiality, integrity, availability — the trifecta of computer security properties
- Learn a general structure for risk assessment

Security Models

The adversarial mindset

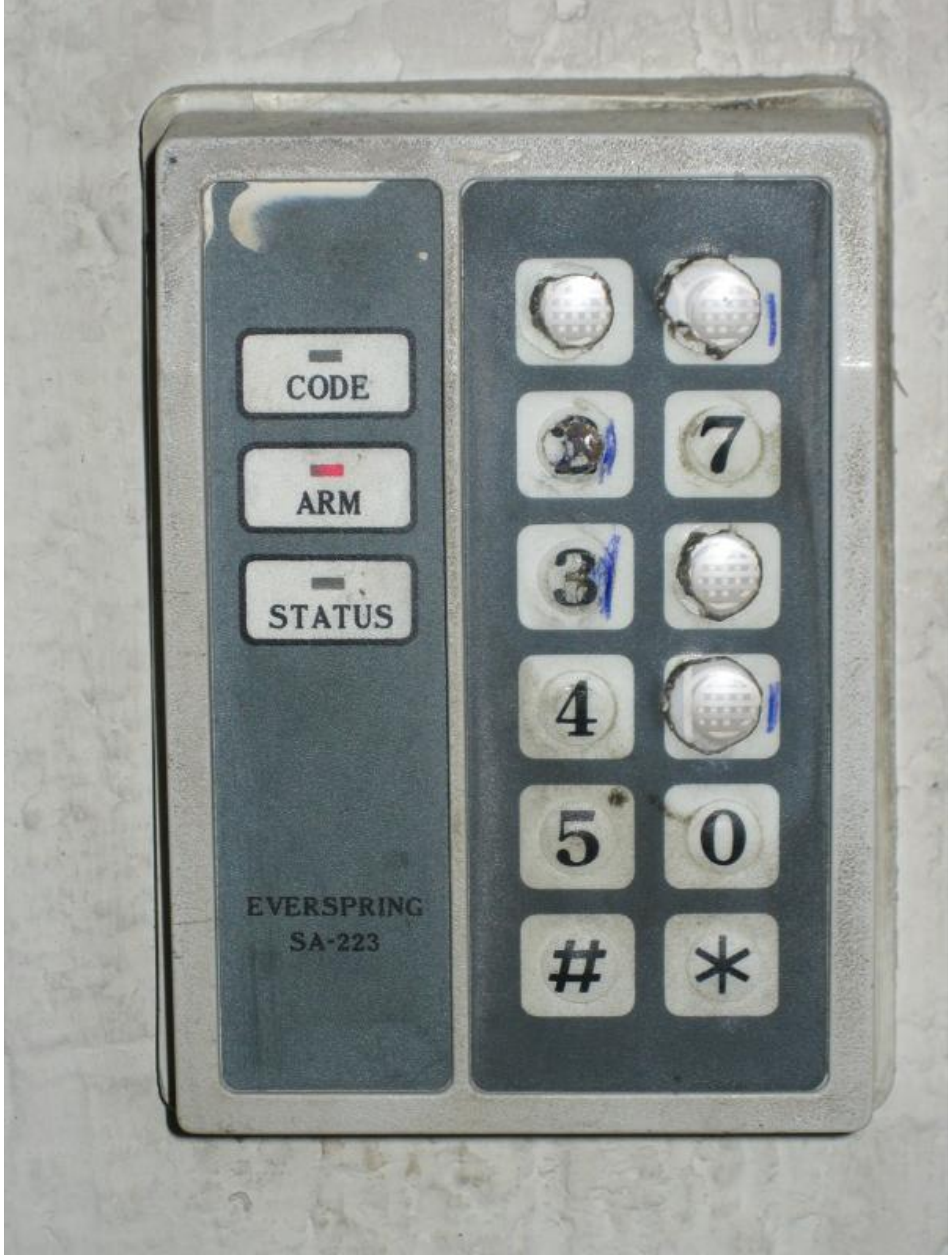
- To build systems hardened against threats… you need to learn to think like an attacker
    - Let’s say you’re playing a game a tag with ten others, and you’re it. What are your strategies for winning as the “attacker?”

The adversarial mindset

- To build systems hardened against threats… you need to learn to think like an attacker
    - Let’s say you’re playing a game a tag with ten others, and you’re it. What are your strategies for winning as the “attacker?”
- Attacker mentality includes…
    - Looking for the weakest links (find the slowest people)
    - Identifying the **assumptions** that proper functionality depends on. Can you make them false? (it’s 10 - 1. Can you make it 9 – 2?)
    - Think outside the box… ignore the limited worldview of the system’s designers (turn tag into a stealth mission)

The adversarial mindset

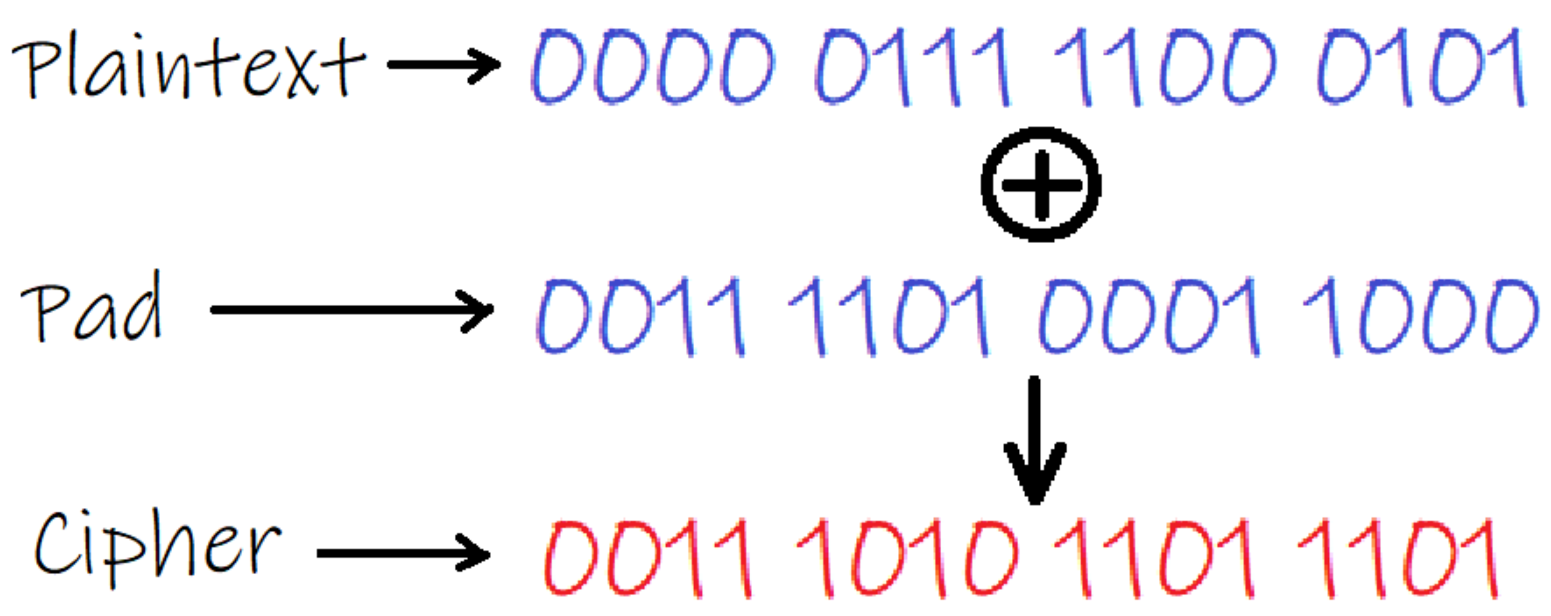
- To build systems hardened against threats… you need to learn to think like an attacker
    - Let’s say you’re playing a game a tag with ten others, and you’re it. What are your strategies for winning as the “attacker?”
- Attacker mentality includes…
    - Looking for the weakest links (find the slowest people)
    - Identifying the **assumptions** that proper functionality depends on. Can you make them false? (it’s 10 - 1. Can you make it 9 – 2?)
    - Think outside the box… ignore the limited worldview of the system’s designers (turn tag into a stealth mission)
- You can do this now and all the time. When you interact with a system, think about what that system depends on and how it might be exploited


CODE
ARM
STATUS
EVERSPRING
SA-223
7
3
4
5
0
#
*

Two competing philosophies for security

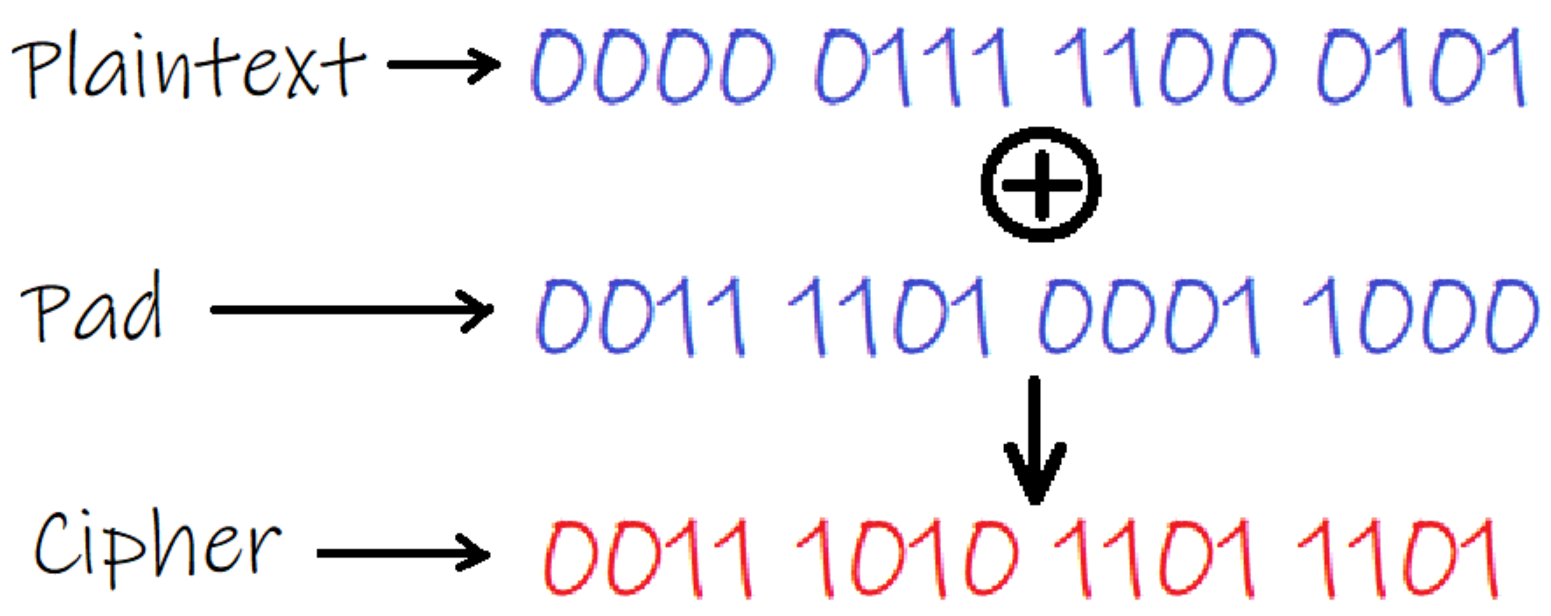
- **Binary** model [secure vs. insecure]
  - Traditional cryptography and trustworthy systems
  - Assume adversary limitations X and define security policy as Y
  - If Y cannot be violated without needing X then system is secure, else insecure
  - Code words: “Proof of security,” “Secure by design,” “Trustworthy systems”
- **Risk management** model [more secure vs. less secure]
  - Most commercial software development (and real-world security… e.g., terrorism)
  - Try to minimize biggest risks and threats
  - Improve security where most cost effective
  - Code words: “Risk,” “Mitigation,” “Defenses,” “Resilience”

Binary model example: Perfect substitution cipher


Plaintext → 0000 0111 1100 0101
⊕
Pad → 0011 1101 0001 1000
↓
Cipher → 0011 1010 1101 1101

- For a given plaintext, choose a string of **random** bits the same length of the plaintext, XOR them to obtain the ciphertext

Binary model example: Perfect substitution cipher



Plaintext → 0000 0111 1100 0101

⊕

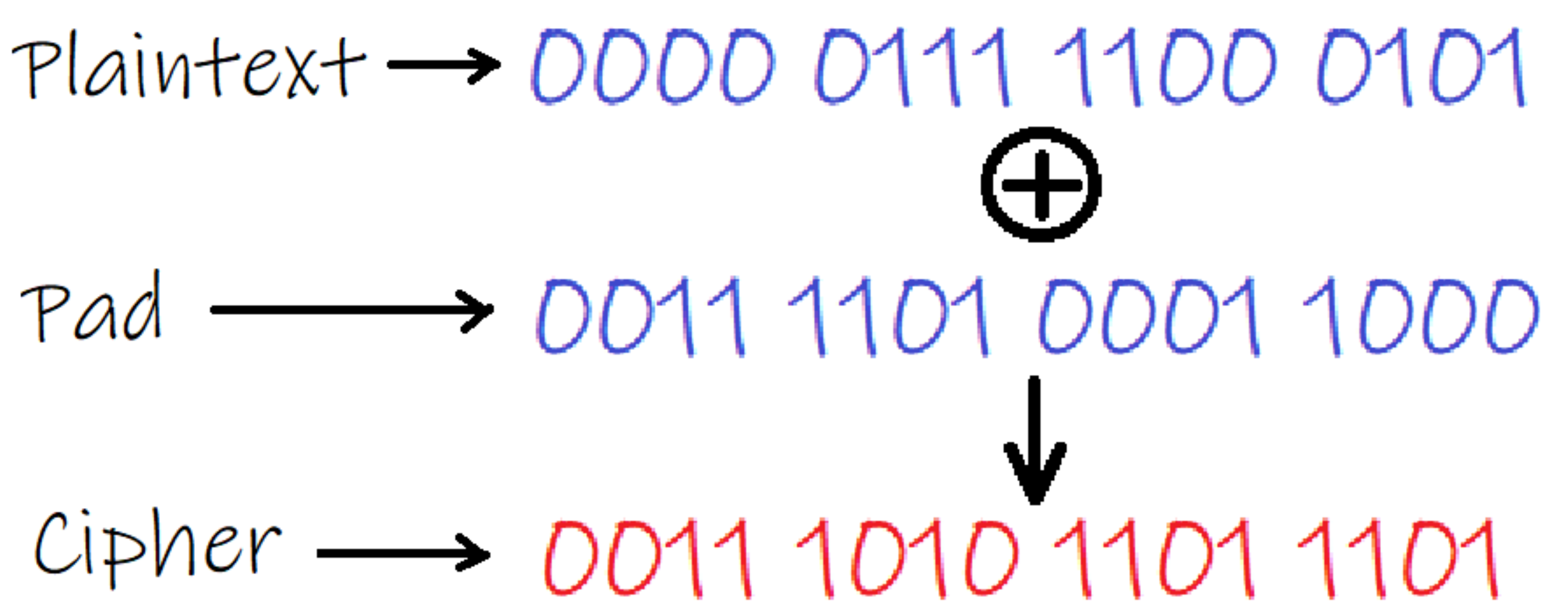
Pad → 0011 1101 0001 1000

↓

Cipher → 0011 1010 1101 1101

- For a given plaintext, choose a string of **random** bits the same length of the plaintext, XOR them to obtain the ciphertext
    - *Why is this considered perfect?*

Binary model example: Perfect substitution cipher



Plaintext → 0000 0111 1100 0101

⊕

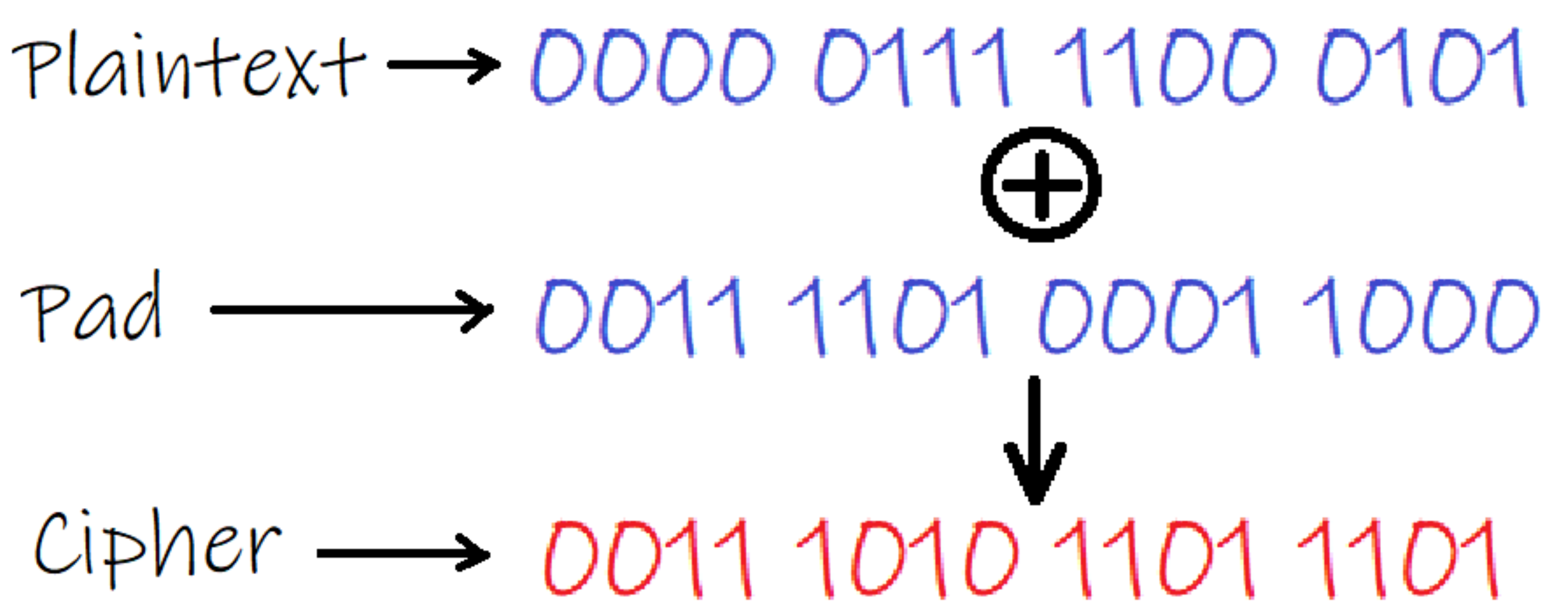
Pad → 0011 1101 0001 1000

↓

Cipher → 0011 1010 1101 1101

- For a given plaintext, choose a string of **random** bits the same length of the plaintext, XOR them to obtain the ciphertext
  - *Why is this considered perfect?*
- **Perfect secrecy** – probability that a given message is encoded in the ciphertext is unaltered by knowledge of the ciphertext
- **Forward secrecy** – Future messages encrypted in this scheme will not reveal information about previous plaintext messages

Binary model example: Perfect substitution cipher


Plaintext → 0000 0111 1100 0101
⊕
Pad → 0011 1101 0001 1000
Cipher → 0011 1010 1101 1101

What are some assumptions about perfect substitution ciphers that might make this not as perfect as it seems?

Problems with Binary: Assumptions often fail in practice

- Many assumptions are **brittle** in real systems
    - Real artifacts are fragile, imperfect, have bugs/limitations
    - *How can you ensure you always generate a truly random one-time pad?*
        - Turns out this is *really hard to do* – we'll talk about failure modes here when we talk about cryptography in weeks 8 + 9
- Often an enormous gap between abstraction and implementation
    - E.g., Randomness in the abstract never goes the way you planned.
    - **Deepak's version:** *The real world is hard.*

Problems with Binary: Security evolution

- As engineers, we like to pretend like we understand our own creations, or that we can create complex systems that only do what they're meant to do…
    - This is a lie, nobody *really* knows how these systems work
    - Complexity of computer systems is approaching complexity of biological organisms
        - 3B base pairs in human genome, 19B transistors in A17 Pro chip…
        - Even more complex with LLMs + modern AI :)
- Complex systems co-evolve with attacks against them
    - Systems deemed secure today may not be resilient to new threats: e.g., quantum computers

Risk-mitigation model example: Antivirus

- Antivirus is software that you install on your machine that monitors your machine to detect + remove **malware** or other bad software

- *Question: What's the difference between different anti-virus software?*

Risk-mitigation model example: Antivirus

- Antivirus is software that you install on your machine that monitors your machine to detect + remove **malware** or other bad software

- *Question: What's the difference between different anti-virus software?*

- Answer: ¯_(ツ)_/¯

Risk-mitigation model example: Antivirus

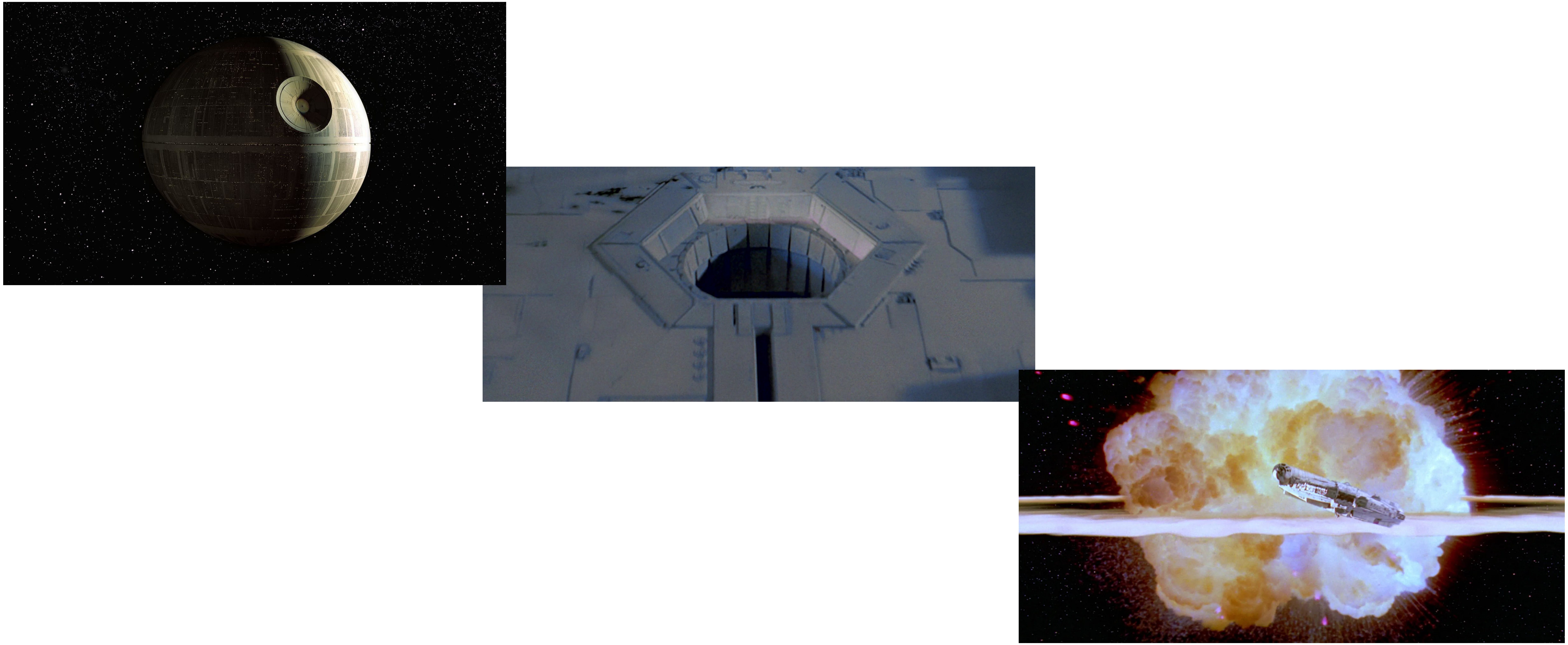
- Antivirus is software that you install on your machine that monitors your machine to detect + remove **malware** or other bad software
- *Question: What's the difference between different anti-virus software?*
- Answer: ¯_(ツ)_/¯
- US Gov't spends ~13B on cybersecurity… often on dozens of products that all do the same thing

Problems with Risk-mitigation

One unforeseen vulnerability can matter ***a lot***

Problems with Risk-mitigation

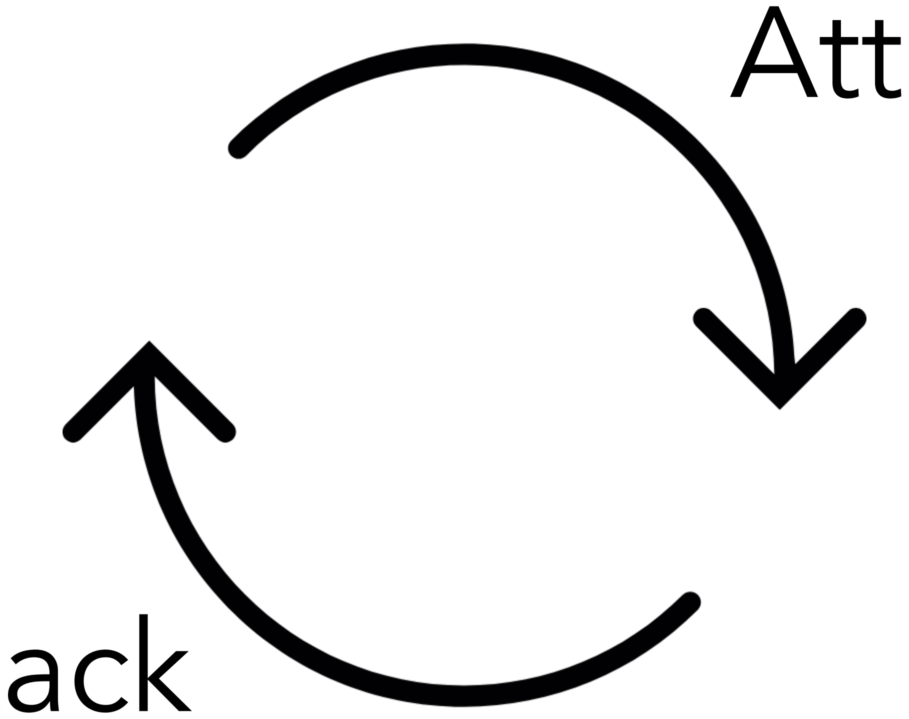
One unforeseen vulnerability can matter ***a lot***



Problems with Risk-mitigation

You never win

- Created arms-race – forced co-evolution



Attacker creates new attack

Defender defends against attack

- Best outcome might just be…. **stalemate**

Problems with Risk-mitigation

How do you know if you're making progress?

- How do you **evaluate** risk or reward?
  - How many "points" of security does antivirus give you? How do you measure those points?
- Big, existential question for the field: how do we measure security?
  - How do we do this in other fields? Are those strategies applicable here too?

Key meta-issues in security

- Policy – what makes a thing bad?
- Assets, Risks, Threats – what do I care about protecting, against what?
- Value – what's the cost if the bad thing happens? how much does it cost to prevent?
- Protection – *how* do I defend against threats? (this is where most of the action is in security field)
- Deterrence – how might I *deter* the bad thing from happening in the first place?

Threat Models

Threat model

Thinking attacker and defender

- A threat model defines security goals: what are we trying to protect and from whom?
  - Threat models are about *assets* and *attackers*
- How can we think about protecting assets?
  - Three properties: *Confidentiality*, *Integrity*, and *Availability* (C.I.A.) of a particular system
- How can we think about characterizing attackers?
  - Who is the attacker? A curious classmate reading your texts over your shoulder?
  - Two properties: *Capabilities* and *Intent*

Threat model

Thinking attacker and defender

- A threat model defines security goals: what are we trying to protect and from whom?
  - Threat models are about *assets* and *attackers*
- How can we think about protecting assets?
  - **Three properties: *Confidentiality*, *Integrity*, and *Availability* (C.I.A.) of a particular system**
- How can we think about characterizing attackers?
  - Who is the attacker? A curious classmate reading your texts over your shoulder?
  - Two properties: *Capabilities* and *Intent*

Trifecta of security: Confidentiality, Integrity, Availability

- What do each of these mean?
  - What is confidentiality?
  - What is integrity?
  - What is availability?
  - What are some examples of each?

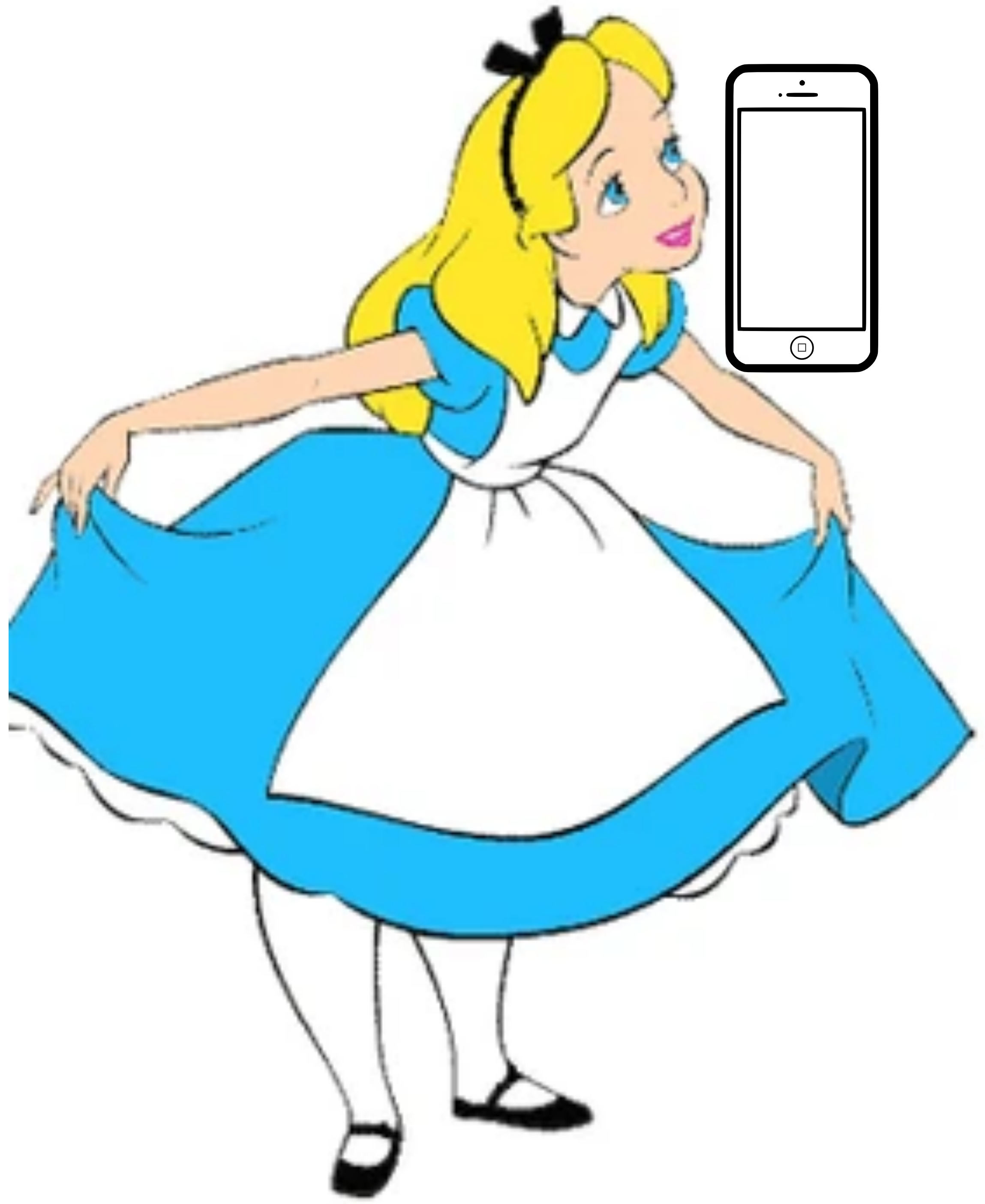
Trifecta of security: Confidentiality, Integrity, Availability

- What do each of these mean?
  - **What is confidentiality?**
  - What is integrity?
  - What is availability?
  - What are some examples of each?

Confidentiality

- Prevention of unauthorized access to information
  - Unauthorized parties can’t view protected information…. in other words, *secrecy*
- What are some examples of breaches in confidentiality?

How does confidentiality work in practice?



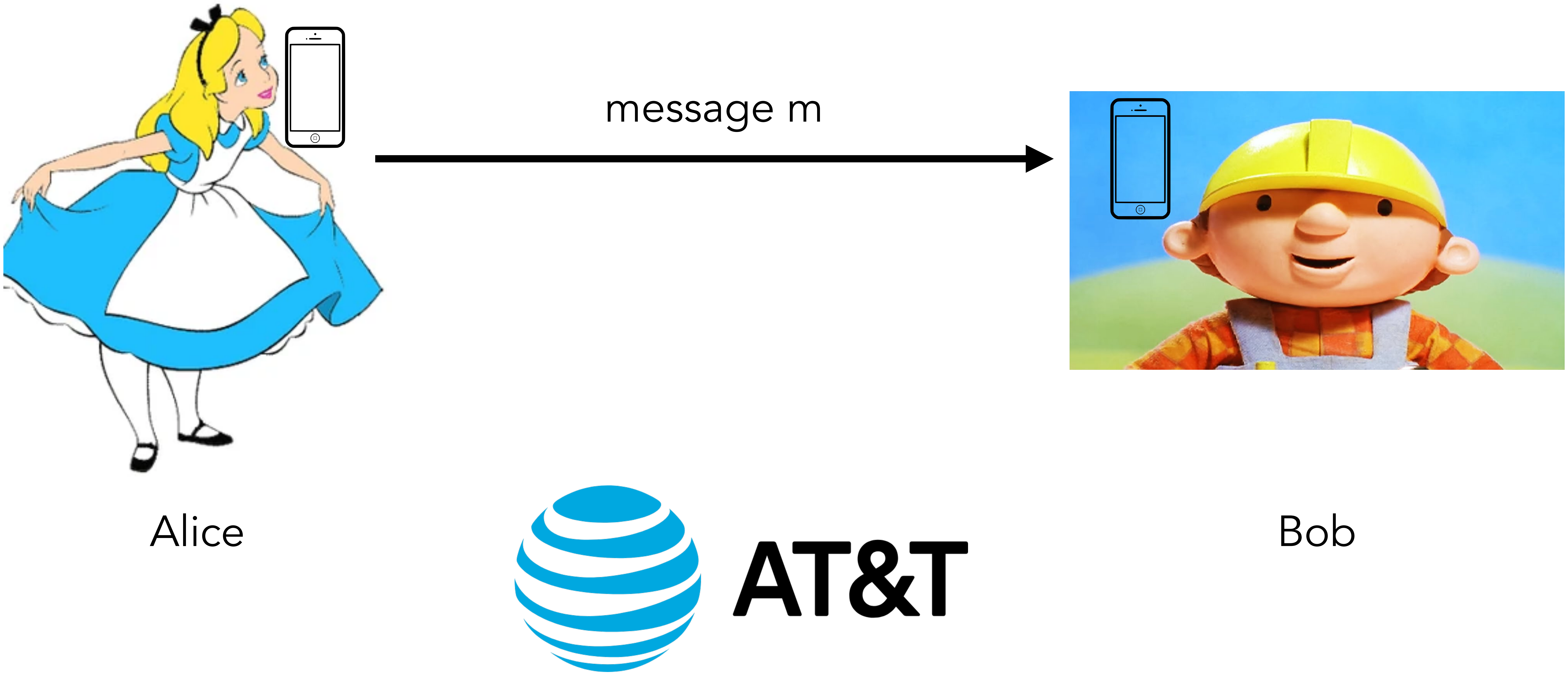
Alice

Bob

AT&T

ISP transmitting message

How does confidentiality work in practice?



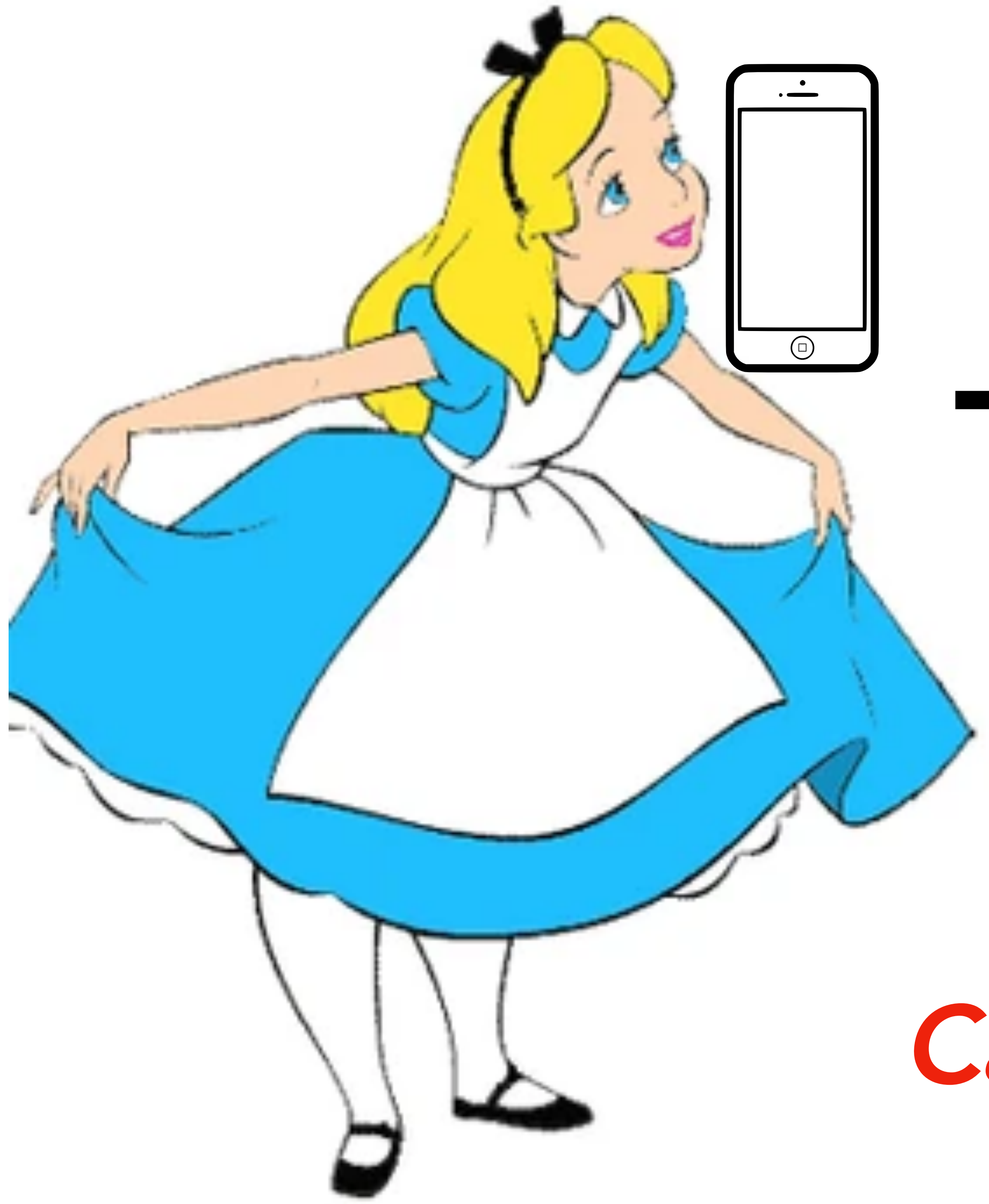
message m

Alice

Bob

ISP transmitting message

How does confidentiality work in practice?



message m

Can AT&T read Alice's message to Bob?

Alice

AT&T

Bob

ISP transmitting message

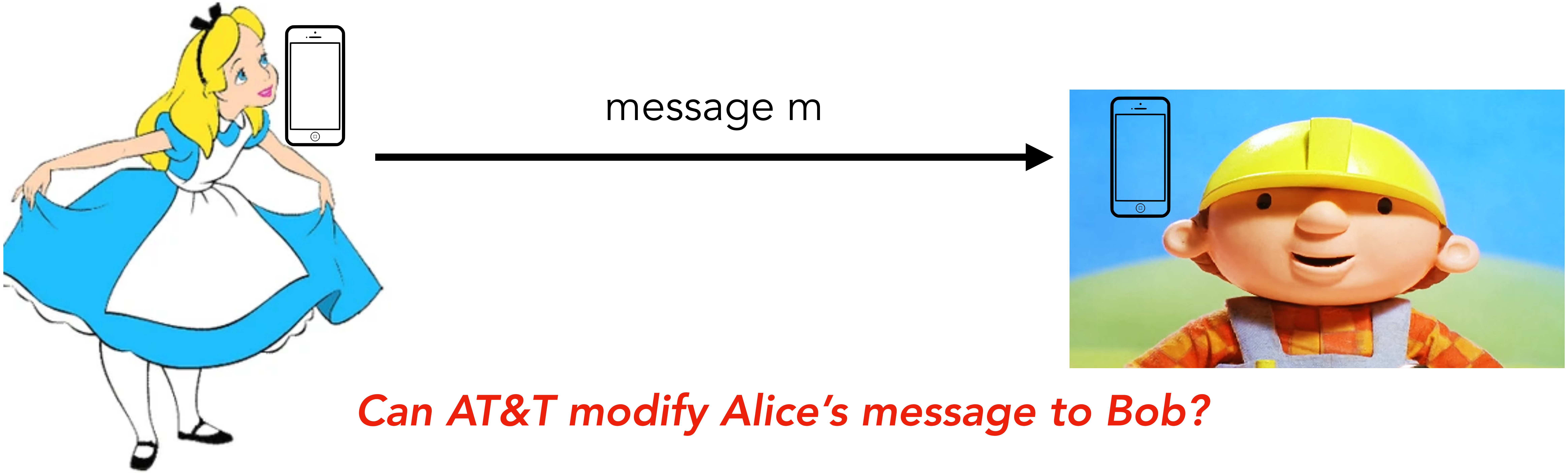
Trifecta of security: Confidentiality, Integrity, Availability

- What do each of these mean?
  - What is confidentiality?
  - **What is integrity?**
  - What is availability?
  - What are some examples of each?

Integrity (& Authenticity)

- Prevention of unauthorized modification of information, process, or function
  - Unauthorized parties can’t modify protect information in flight or at rest
- What are some examples of breaches in integrity?

How does integrity work in practice?



Can AT&T modify Alice's message to Bob?

Alice

AT&T

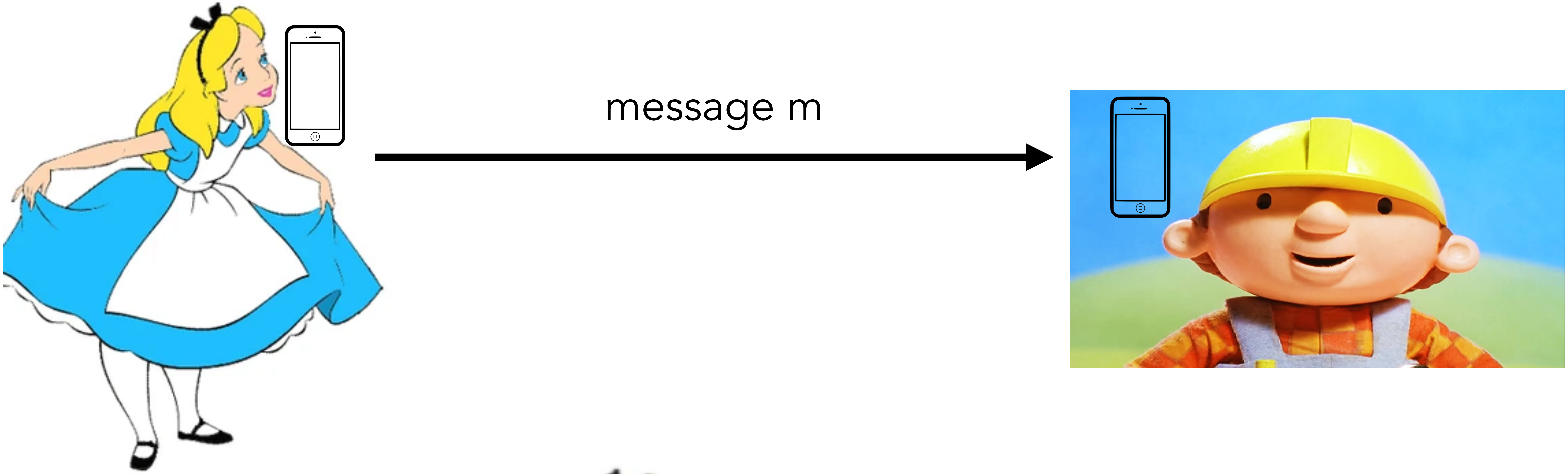
Bob

ISP transmitting message

Integrity (& Authenticity)

- Prevention of *impersonation* of another identity… like integrity, but specifically to do with another *actor* (person, system, otherwise)
- What are some examples of breaches in authenticity?

How does authenticity work in practice?



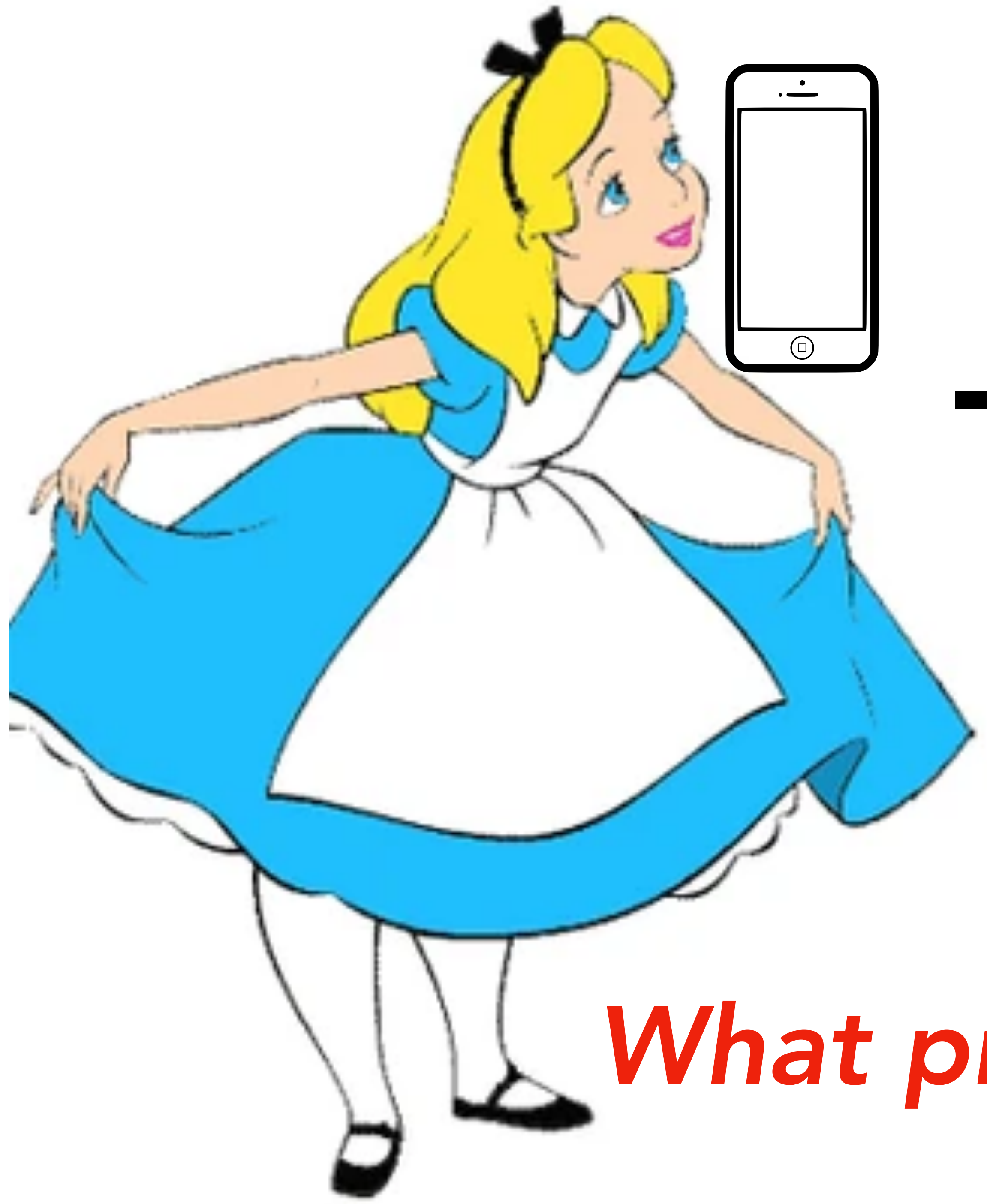
message m

Alice

Bob

Totally 100% Alice

How does authenticity work in practice?



message m →

What prevents AT&T from masquerading as Alice?

Alice

Bob

Totally 100% Alice

Trifecta of security: Confidentiality, Integrity, Availability

- What do each of these mean?
  - What is confidentiality?
  - What is integrity?
  - **What is availability?**
  - What are some examples of each?

Availability

- Prevention of unauthorized *denial of service* to others
  - Unauthorized parties can’t prevent authorized users from accessing a system
- What are some examples of breaching *availability*?

How does availability work in practice?

message m

Can AT&T block Alice's message to Bob?

Alice

Bob

AT&T

ISP transmitting message

C.I.A. + Privacy

- **Privacy:** A person's right or expectation to control the disclosure of their personal information, **<u>including</u>** activity metadata

- What is the difference between privacy and secrecy?

C.I.A. + Privacy

- **Privacy:** A person's right or expectation to control the disclosure of their personal information, **<u>including</u>** activity metadata
- What is the difference between privacy and secrecy?
  - Secrecy is about explicitly hiding information from third-parties
  - Privacy is about not being observed / monitored, including public data
- Activity metadata
  - What can you figure out about a person just from their location history?

C.I.A. + Privacy

- What security property is violated if someone…
  - Unplugs your alarm clock while sleeping?

C.I.A. + Privacy

- What security property is violated if someone…
  - Unplugs your alarm clock while sleeping?
  - Changes the time on your alarm clock?

C.I.A. + Privacy

- What security property is violated if someone…
  - Unplugs your alarm clock while sleeping?
  - Changes the time on your alarm clock?
  - Watches you through your window via a telescope?

Vulnerabilities

- Weakness that can be ***exploited*** (made use of) to cause damage to assets (usually in the form of a violation of C.I.A. + Privacy)
  - Default passwords (e.g., “admin123”)
  - Bad passwords (e.g., “password123”)
  - Implementation flaws in software
  - Old software left open to the network
  - Cryptography based on weak keys
- Lots of security is organized around vulnerabilities (e.g., National Vulnerability Database, run by NIST), always buzz when a **0-day vuln** is released

Threat model

Thinking attacker and defender

- A threat model defines security goals: what are we trying to protect and from whom?
  - Threat models are about *assets* and *attackers*
- How can we think about protecting assets?
  - Three properties: *Confidentiality*, *Integrity*, and *Availability* (C.I.A.) of a particular system
- How can we think about characterizing attackers?
  - **Who is the attacker? A curious classmate reading your texts over your shoulder?**
  - **Two properties: *Capabilities* and *Intent***

Attackers / Adversaries

Know thine enemy

- Understand who the attacker is
  - Individual? Are they an outsider, insider, or privileged insider?
  - Group? Are they ad hoc? Established hacking group?
  - Organization? Are they a competitor? A supplier? A customer?
  - Nation state? How powerful is the nation state?

Capabilities

- Clearly understanding the capabilities and scope of the attacker is crucial to proper security defense
- What are some other capabilities of attacker that might be useful to know about?

Motivation / Intent

- Motivation plays a big role in our lives
  - Would you rather fight a motivated 5-year old or an unmotivated 35-year old?
- What are some motivations attacker might have?

Trusted Computing Base – defines your trust

- **Trusted Computing Base** (TCB)
  - Set of systems/components/people/entities that your security depends on
- Remember from last time…
  - You *need* to trust something… so better to define what you choose to trust
  - Truth is objective, Trust is relative

plate as TCB

Trust/Security Boundary + Attack Surface

- **Security Boundary**
  - Perimeter around components of the same trust level
  - Any data or signals coming in from outside is untrusted and potentially malicious (e.g., a bouncer)
- **Attack Surface**
  - Set of interaction points across a security boundary
  - Parts of your system handling input from or otherwise interacting with less trusted and potentially malicious entities
  - Some *highly sensitive* systems are even “air-gapped” to minimize the attack surface

Back to threat models…

- Your very first question in any security discussion should be

What's the threat model?

- You can't argue about attacks or defenses without understanding the threat model
- The threat model is what *defines* the problem to be solved
  - And if there's no consensus on the problem, there's going to be no consensus on the solution

Threat Modeling Exercise: Breaking into CSE after hours

You are trying to keep the riff raff out

- Assets?
  - What are you trying to protect? What’s at risk?
- Attackers?
  - What are their capabilities?
  - What are their motivations?
- What’s your security boundary? What’s the attack surface?

A word of caution about threat models

- Your threat model is <u>your</u> problem scope… attackers do not care about them
- Just because an attacker doesn't exist in your threat model doesn't mean they don't exist :)
    - It just means you have explicitly decided you will not address them in your solution
- *"All models are wrong, but some are useful" – George E.P. Box*

Quick Break

Risk Assessments

Risk Assessment

- Security is *rarely* binary, everything has some measure of risk
  - Risk: Very low to Very high
- Calculated as some combination of:
  - Likelihood — the probability the security threat will materialize
  - Impact — what will happen if the threat actually takes place
- We evaluate security risk relative to value
  - Is this risk worth taking?

Risk Assessment (in a perfect world)

1. Understand system requirements
2. Identify assets + attackers
3. Establish security requirements
4. Evaluate system design
5. Identify threats and classify risks
6. Address identified risks

1. Understand system requirements

- What is the system actually supposed to do?
    - What are the inputs and outputs of the system?
    - What is the security boundary?
    - Is there anything vague about this boundary specification?
    - How is the system going to be deployed? How long is it deployed for?

2. Identify assets and attackers

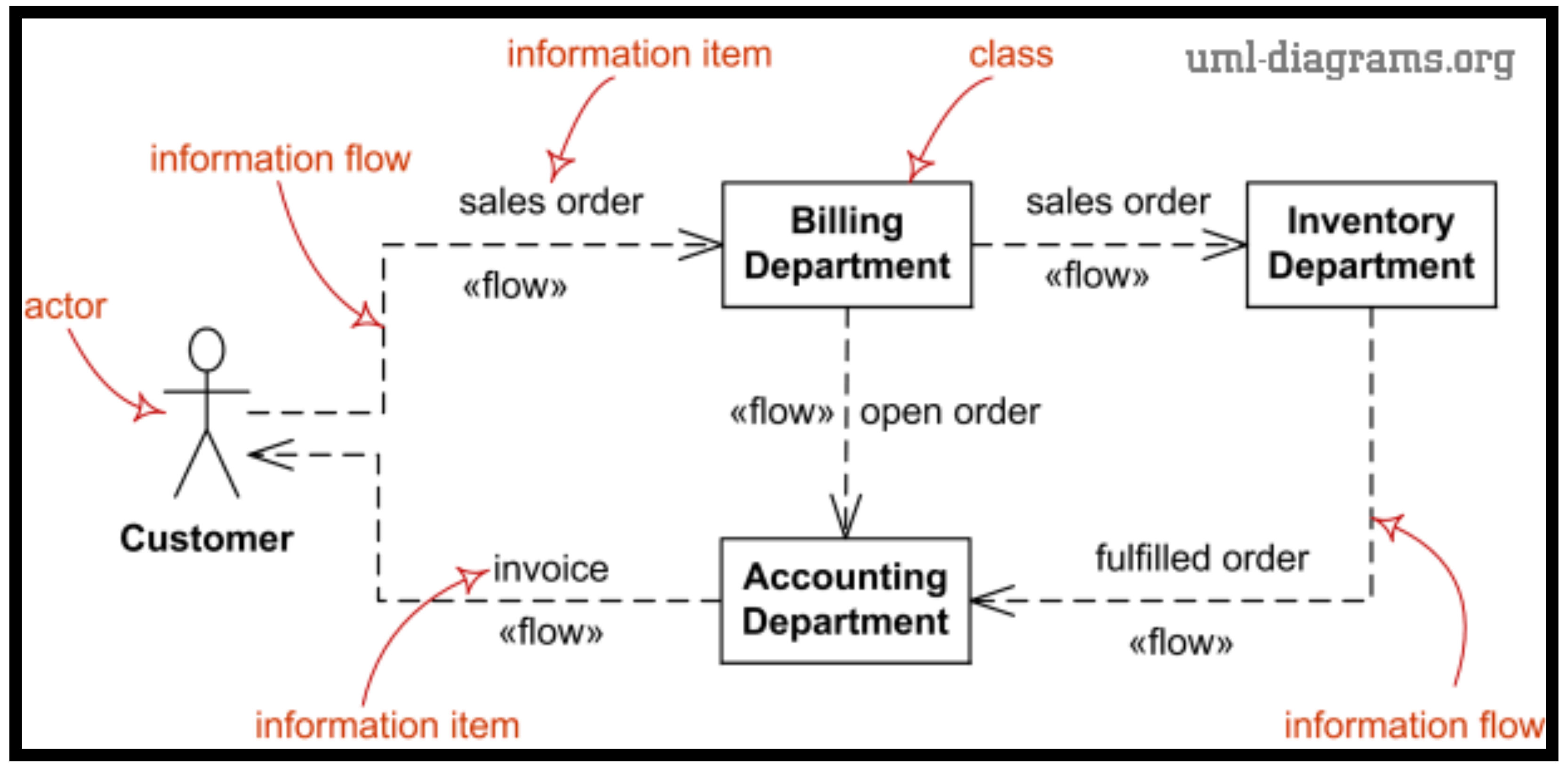
- Who are the stakeholders?
  - Who has a vested interest in the system running correctly / securely?
- What needs to be protected?
  - And how much is it worth? To whom is it worth that much?
- From whom does it need protection?

3. Establish security requirements

- Think about assets and C.I.A + Privacy…

4. Review system design

- Really, now, go understand how the system works
- What are the different components in the system?
    - How do the components communicate?
    - How will they store and/or pass data?
- Draw an *information flow diagram*
    - Look for ambiguities
- The design will evolve as risks are identified and mitigated, so need to continuously monitor changes



5. Identify threats and classify risks

- Using the security goals and current design, identify **threats** to security
- For each threat, determine likelihood and convert to a “risk score”
    - It’s obviously sort of ad-hoc, but it’s the best we know how to do…
- Risk assessment is subjective
    - But your implicit knowledge (of a system), common sense, and rational paranoia will help you make sure the risk ranking is reasonable

6. Address risks

What do you do about risks?

- **Avoid:** Remove the component that creates the risk
  - E.g., remove a feature from a product
- **Mitigate:** Add measures that decrease the likelihood or impact
  - E.g., Build a defense or wall around your riskiest component
- **Transfer:** Make it someone else's problem
  - E.g., buy insurance
- **Accept:** Do nothing
  - Risk is never truly 0, you always accept *some* risk…

Risk acceptance

- You can't get anything done or offer anything without some risk
- *Residual risk* is what is left over when all the controls and protections have been installed
- *Remember: accepting the risk does not make it go away.*

Additional resources

- Adam Shostack's excellent book *Threat Modeling: Designing for Security*
    - *https://threatmodelingbook.com*
- *NIST Guide for Conducting Risk Assessments*

Just to recap…

- To defend systems, you need to adopt an *adversarial mindset*
- ***Threat models*** are mechanisms to evaluate security. What are we trying to protect and from whom?
  - Assets: *Confidentiality, Integrity, Availability*
  - Attackers: *Capabilities* and *Intent*
- Security involves analyzing **risk**
  - Calculated as some combination *of* likelihood and impact
  - Is this risk worth taking?

Next time…

- We get technical…
  - Function calls in C, how return addresses are stored, buffer overflows…
  - Get ready to talk about memory, addresses, registers :)
- PA1 is due 1/15