# Authorization

UT CS361S

FALL 2020

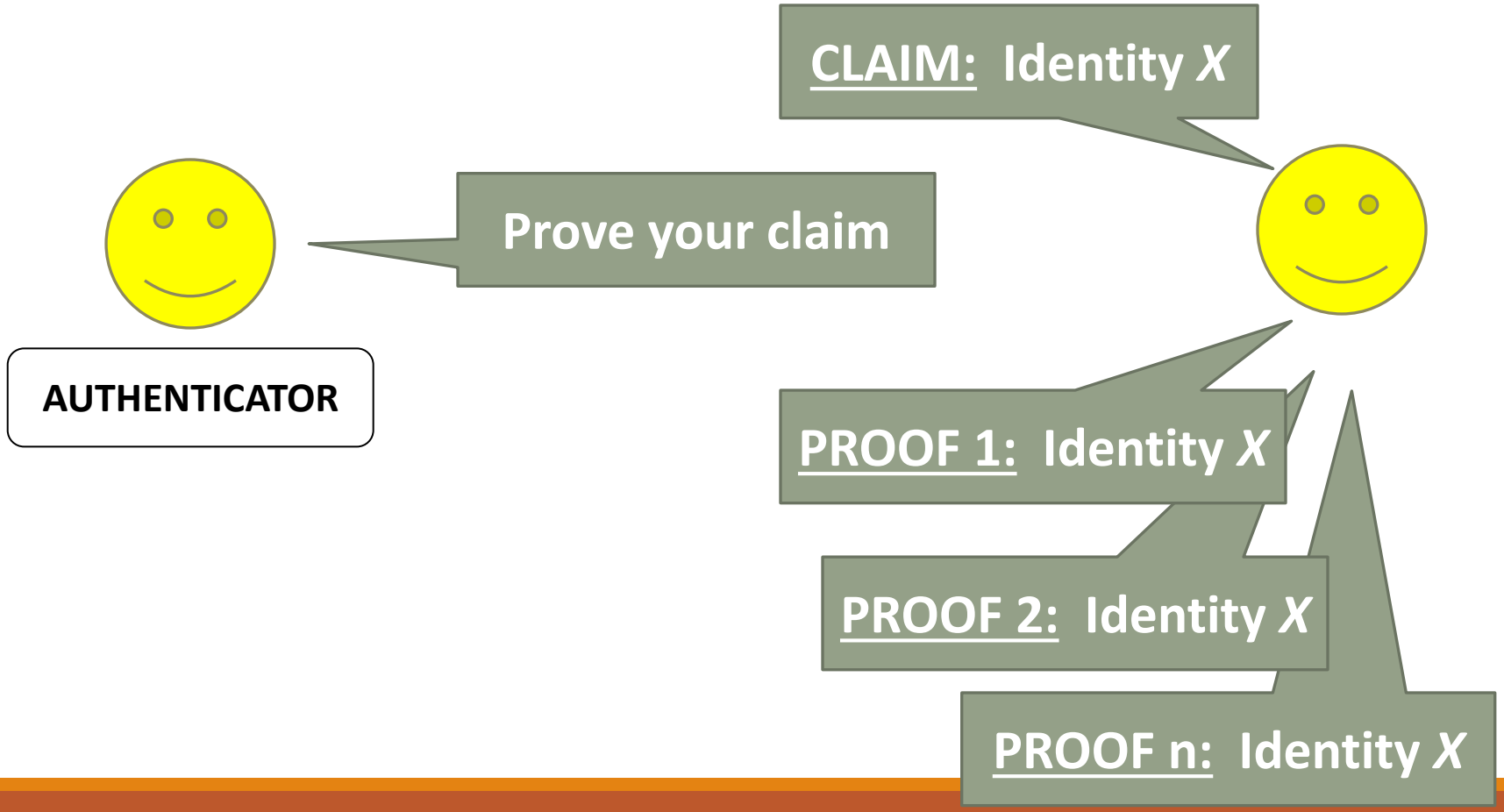
LECTURE NOTES

# Authentication/Authorization

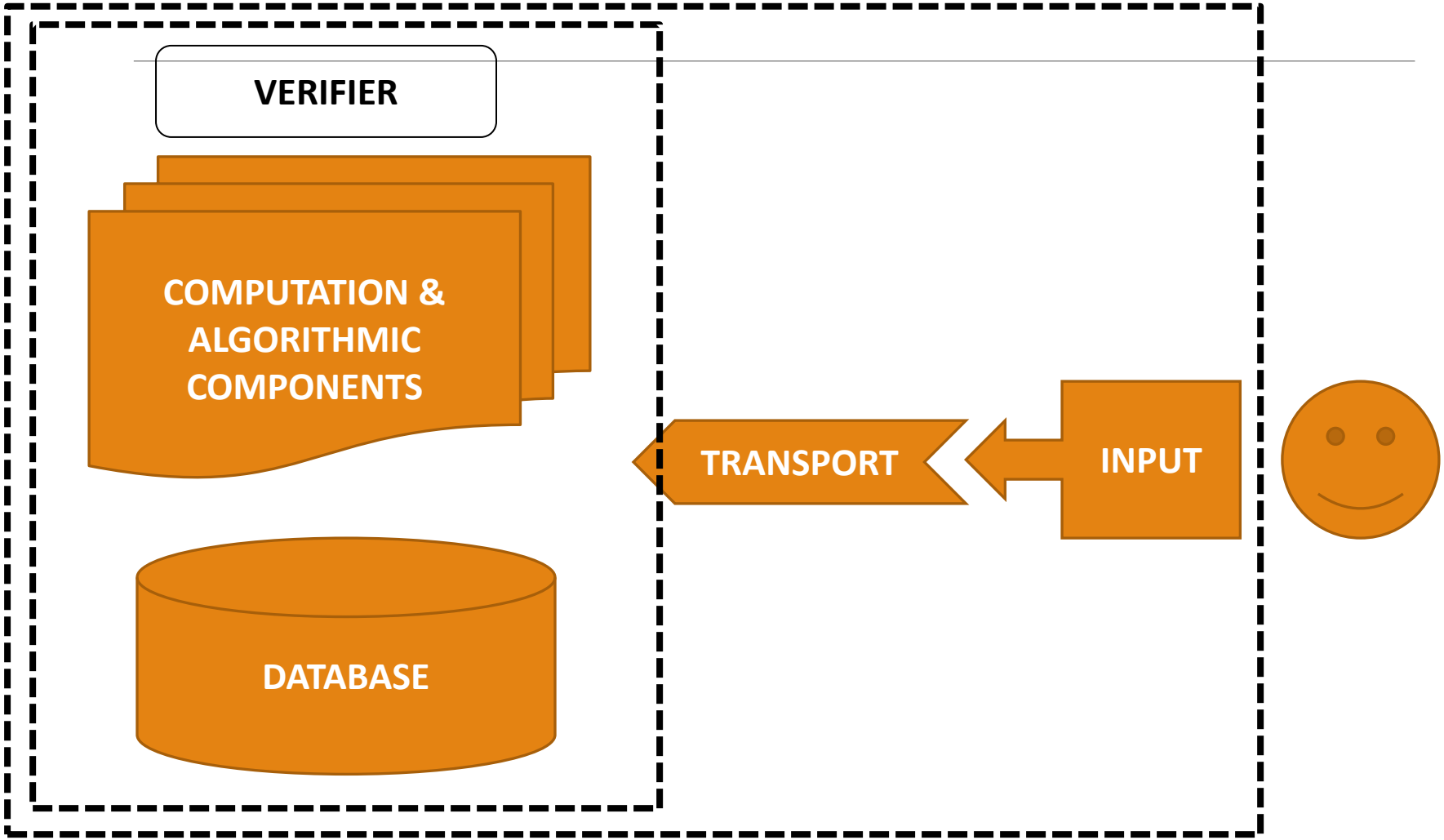
Validating Identity

Permissions Assigned to a Validated Identity

# The Authentication Process



# Authentication Mechanism



# The Big Three

Something you **KNOW**

Something you **HAVE**

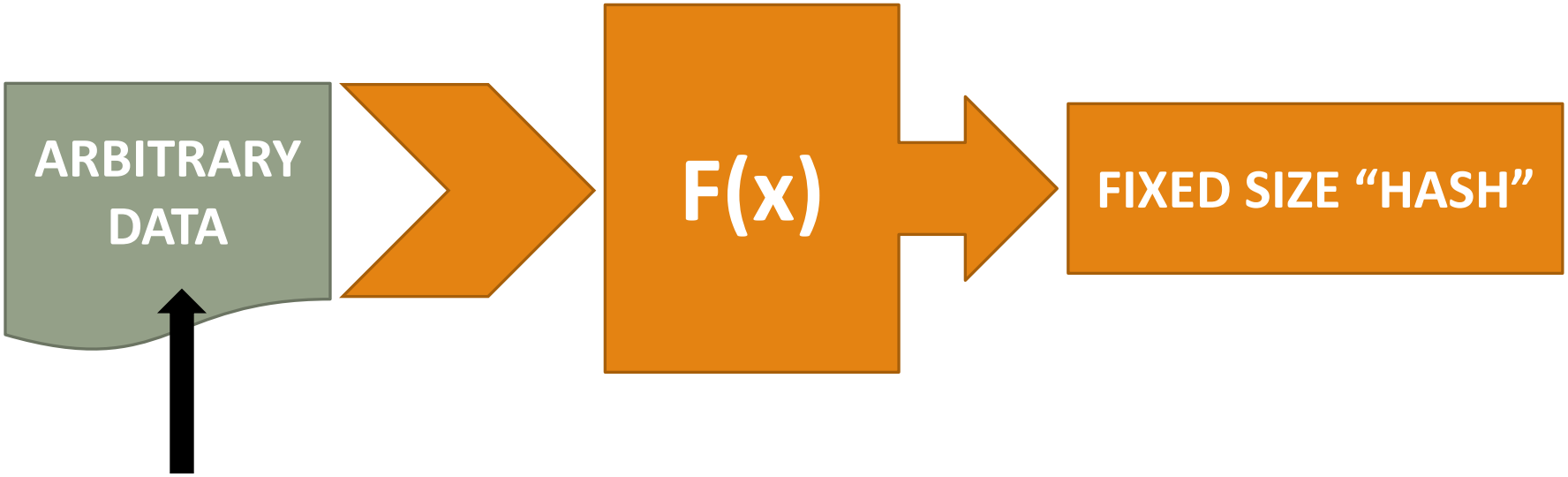
Something you **ARE**

# KNOW: Passwords

Security Requirements

1. The password is ONLY known by the party seeking authentication
2. The password cannot be easily guessed by human or computer
3. The password will not be forgotten by the party seeking authentication

# Sidebar: Hashing




THIS IS CALLED THE “PRE-IMAGE”

# Required Properties

1. “Compression”: The size of x is unbounded; the size of f(x) is fixed
2. “Ease of Computation”: f(x) is “easy” to compute
3. “Preimage Resistance”: Given y, it is “hard” to find x
4. “2nd Preimage Resistance”: Given x and f(x)=y, “hard” to find f(x’)=y
5. “Collision Resistance”: It is hard to find (x, x’) where f(x) = f(x’)

# Simple Intuition

A hash is a small, fixed size mathematical fingerprint of data

You cannot recover ***or predict*** the data given the fingerprint

Any change in the data results in a complete change of the fingerprint

# Common Hash Algorithms

MD5 – DEPRECATED, DON'T USE

SHA1 – DEPRECATED, DON'T USE

SHA256 – Currently Recommended

EXAMPLES:

sha256("hello")

2cf24dba5fb0a30e26e83b2ac5b9e29e

1b161e5c1fa7425e73043362938b9824

# Salted Hashes

Google “5e884898da28047151d0e56f8dc6292773603d0d6aabbdd62a11ef721d1542d8”

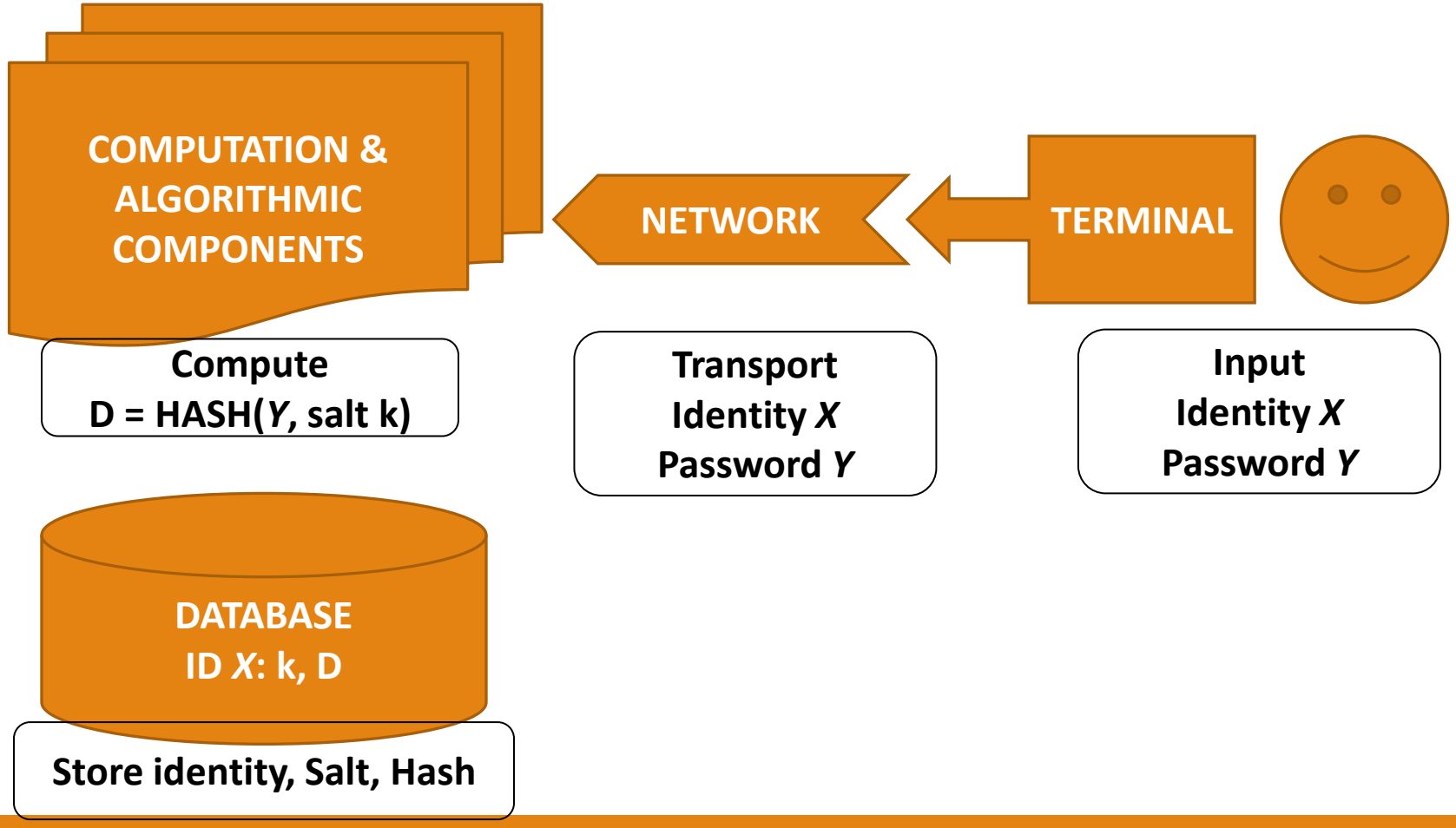
The SHA256 hash of something will be the same until the end of time

If we want the output to be unrecognizable, we can add a “salt”

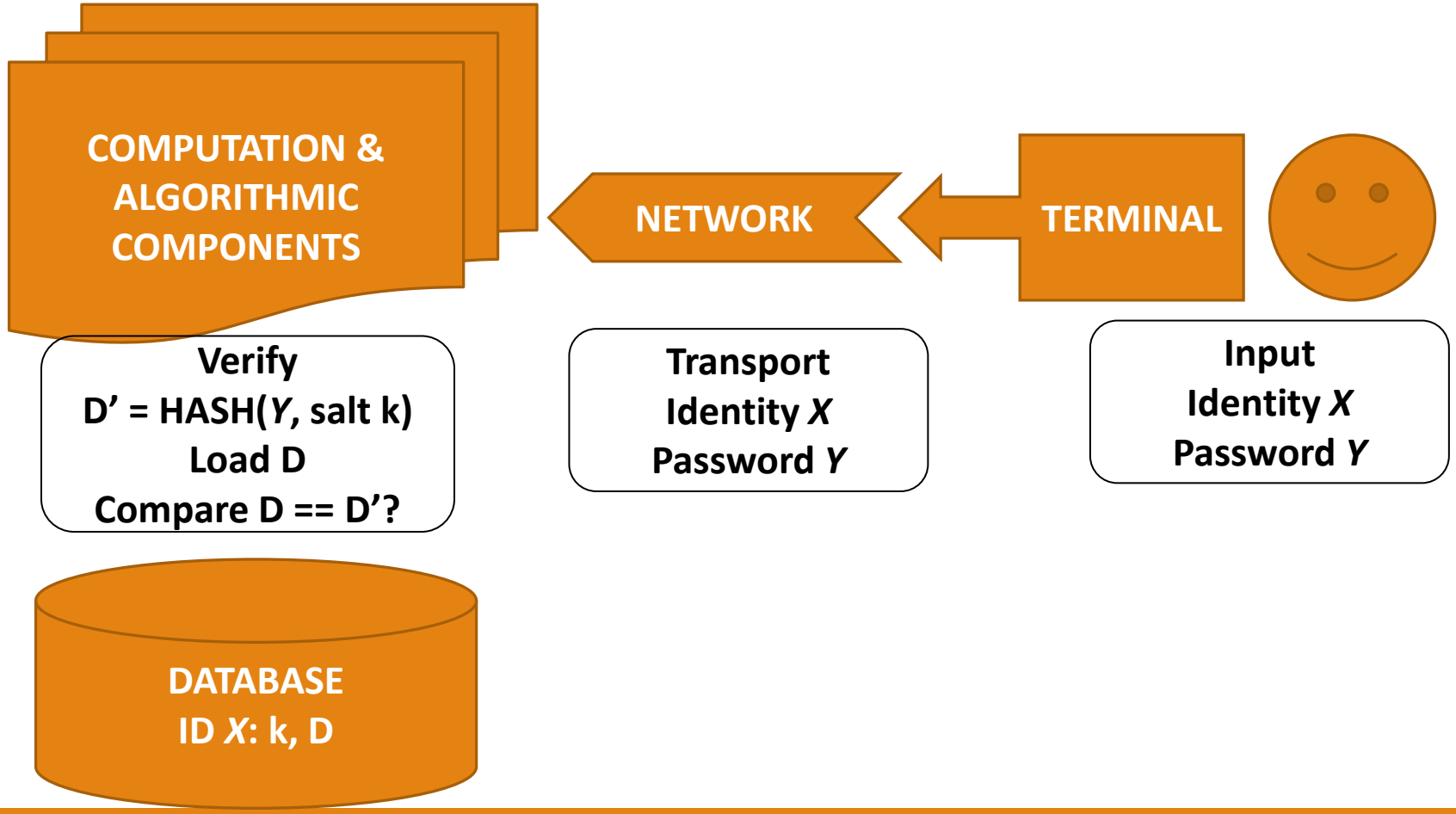
A salt is just some random data. It can be publicly known

SHA256(“hello”+32 bytes of random) = a random value

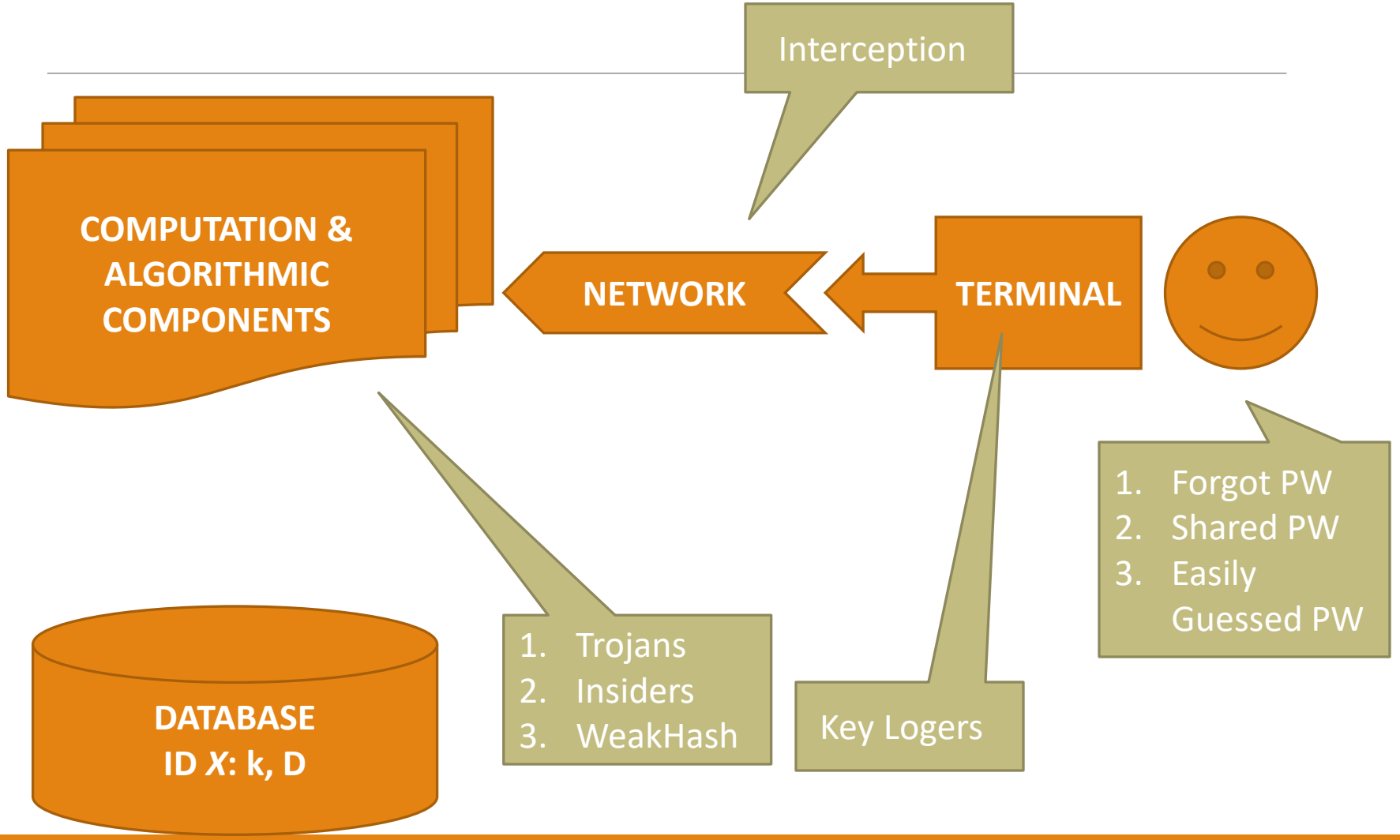
# Password Registration



# Password Verification



# Common Problems

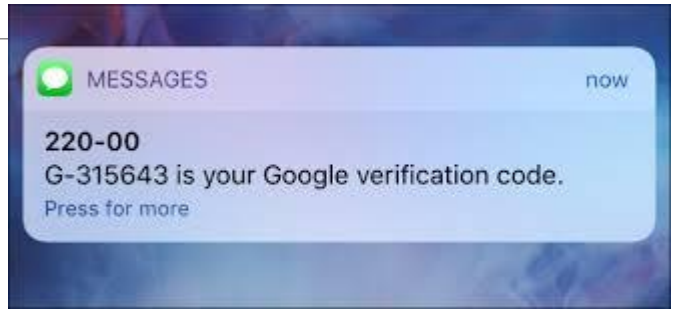


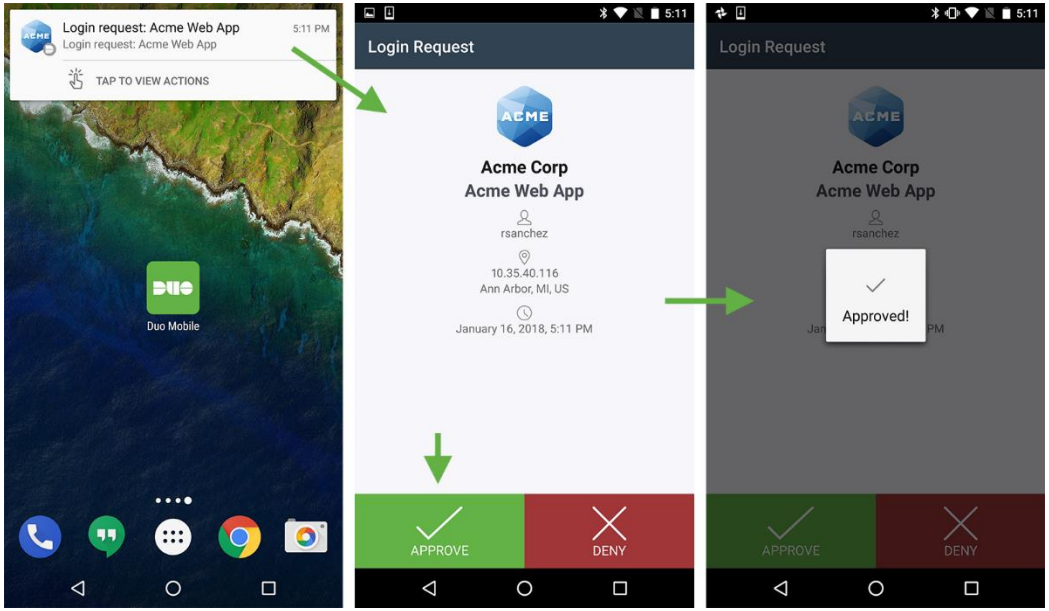
# Something you Have

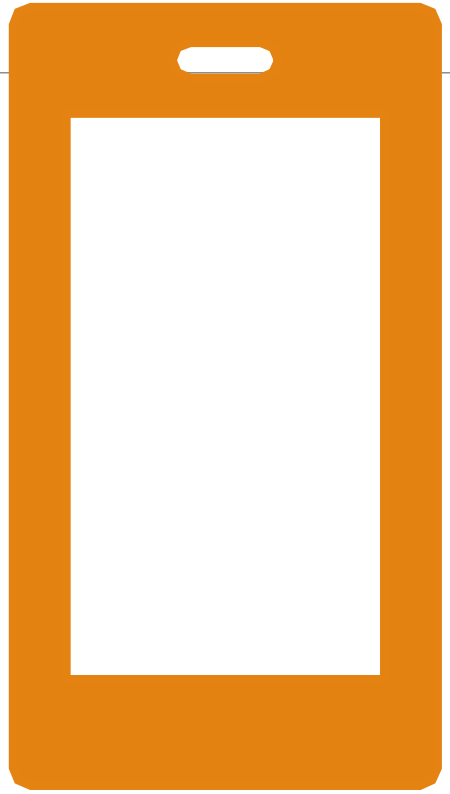
Security Assumptions

1. The “token” is ONLY possessed by the party seeking authentication
2. The token cannot be easily forged or duplicated
3. ***The authentication protocol is secure***

# Something you Have Examples



# Problems with "Tokens"

Is it ***REALLY*** something you have?

Is sending a code by email 2-factor?

What about phone cloning?

What about network interception?

Is an RSA Token's seed just ***something you know***?

"Something you can respond with"

## Security Assumptions

1. The “characteristic” is effectively unique
2. Can effectively measure, record, or detect the characteristic
3. Characteristic cannot be forged, replicated, or otherwise “lost”
4. Characteristic will not change (too much) over time
5. Characteristic will never need to be revoked
6. ***The Authentication Protocol is Secure!***

# Something you Are

# False Positives vs False Negatives

False Negative – Do not authorize party with valid characteristic

False Positive – Authorize party with invalid characteristic

# Receiver Operating Characteristic

The trade off between FP and FN

Decreasing one typically increases the other

Equal Error Rate is when FP approximately equals FN

In most biometrics, ***False Negatives*** are worse

# Problems with Biometrics

1. ~~Fingerprinting has been *seriously* misused in Courts (see Anderson at pp. 469-470)~~
2. ***Interpretation of results and understanding of statistics***
3. Variable accuracy in scanning mechanism
4. “Freshness”
5. Belief in infallibility leads to security culture problems
6. Biometrics exclude a *lot* of people (e.g., differently abled)
7. Cvil Rights and Privacy issues
8. Injury that alter the characteristic (e.g., fingerprint)

# Access Controls

The mechanism by which authorization permissions are managed

Within most information systems, the most common controls:

- (C)reate
- (R)ead
- (U)pdate
- (D)elete

Most other controls can be thought of as a form of one of these

(But “Execute” is often listed separately)

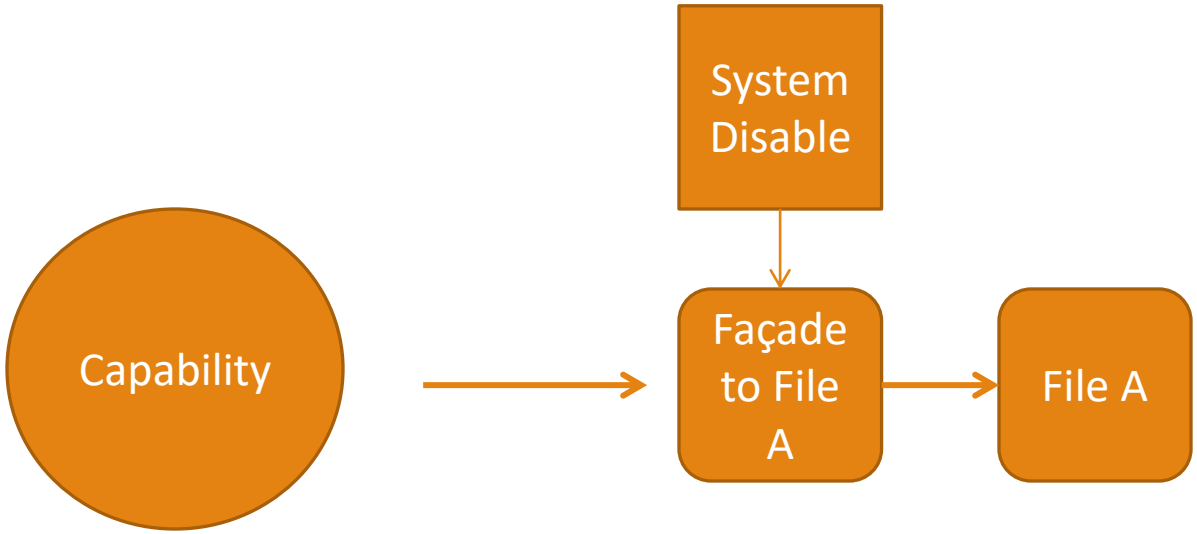
# Every-day Approaches

ACCESS CONTROL LISTS

CAPABILITIES

Opponents of capabilities argue that you cannot change a file's status

They just don't understand capabilities



# MAC vs DAC

Mandatory Access Controls – what is permitted is determined by policy

Discretionary Access Controls – what is permitted is determined by user

# Multi-Level Security (MLS)

Often seen as synonymous with MAC

Users and data are assigned classifications

What users are permitted to do with data depends on both labels

# Bell Lapadula Model

Design emerged from military document classification

Enforces two properties

- *Simple Security Property:* No Read Up (NRU)
- **-Property:* No Write Down (NWD)

The *-property was the big innovation of BLP. It *assumed* trojans and buggy code!

This is a well defined security policy

- It is relatively easy to determine if the mechanisms enforce the policy
- If it's the right policy it works great!

# Problems of BLP

If the security officer can “temporarily declassify” all of the protections go away

- Strong tranquility: security labels never change during operation
- Weak tranquility: labels never change in a way that violates security policy
  - The idea here is “least privilege”. Even if you have TS, start at unclassified
  - As you access info that is higher, your level increases

The system can get fragmented into pieces that can’t communicate

Also, what do you do with an App that has to straddle?

- A document editor used to redact a TS document to Classified

Doesn’t deal with creation of subjects or objects

# Biba model

Upside-down BLP

- You can only read up and write down
- The goal is *integrity* not *confidentiality*

Partially used in Vista. Uses the NoWriteUp.

- Most files are “medium” or higher. IE is “low”
- So, things downloaded can read most files, *but not write to them!*

This was the first formal model of integrity

- Struggled in real-world because of the exceptions and straddling issues

# Inference

Information sharing often involves some kind of “scrubbing”

In MLS, a report is redacted before moving down a security layer

In privacy-preserving systems, data is often *anonymized*

The problem, of course, is inference

- People can often be identified by their medical records even with names removed
- And, of course, we’ve seen this with AOL and Google

# Inference Control

Characteristic formula – the query instructions to get some set

Query set – the set produced by a characteristic formula

Sensitive Statistics – stats that deanonymize information:

- For example, if the set is too small, than we've identified an individual by attributes

# Query Size

You can limit how small a result is from a query

But you also have to worry about returning N-1!!

Also, you have to deal with using multiple queries to get a smaller than N intersection

# Role Based Access Control

- RBAC
- Users assigned roles, permissions based on roles
- Is this MAC or DAC?
- Lessons from the field: what goes wrong in RBAC?

# Authorization Principles

- Least Privilege
- Separation of Duties