Quick note

- Check Piazza
- Part 1 goal:
  - 1. You log in to lnxsrv
  - 2. Use ssh-agent (if you have a passphrase for your key, you can use it here)
  - 3. “ssh __@lnxsrv.seas.ucla.edu” should not ask for password OR passphrase
- More people came to office hours
  - Poll: is 5pm-7pm better than 4pm-6pm?
  - You can also email me
- Reminders:
  - Weekly quiz
  - Assignment 2 is meant to be easy
  - If you expect to be busy soon, start other assignments early!

Feedback / Office Hours

- Tameez Latib
  - tameezlatib@gmail.com, please add “CS35L” to the subject line
  - Office Hours: Monday **4pm-6pm** (or by appointment)
  - Feedback: https://forms.gle/6kcJ2aJtzAzFMhHQ7 (anonymous google form)

Quick refresher

- Asymmetric encryption:
- Bob is sending a message to Alice
  - Bob sends ciphertext=Encrypt(plaintext, __1__) to Alice
  - Alice finds plaintext=Decrypt(ciphertext, __2__) from Bob
- What's (1) ?
- What's (2) ?

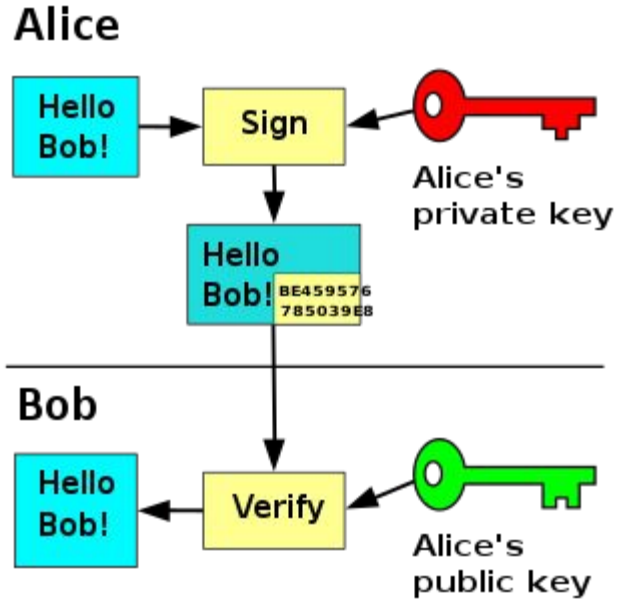
Gpg

- Gnu privacy guard / GnuPG
  - Allows you to encrypt, sign, etc
- SSH sets up a secure tunnel, so that data can flow from A to B
  - Takes care of encryption
- GPG gives the user the power to encrypt
- Check ~/.gnupg, keys are stored here

What are signatures?

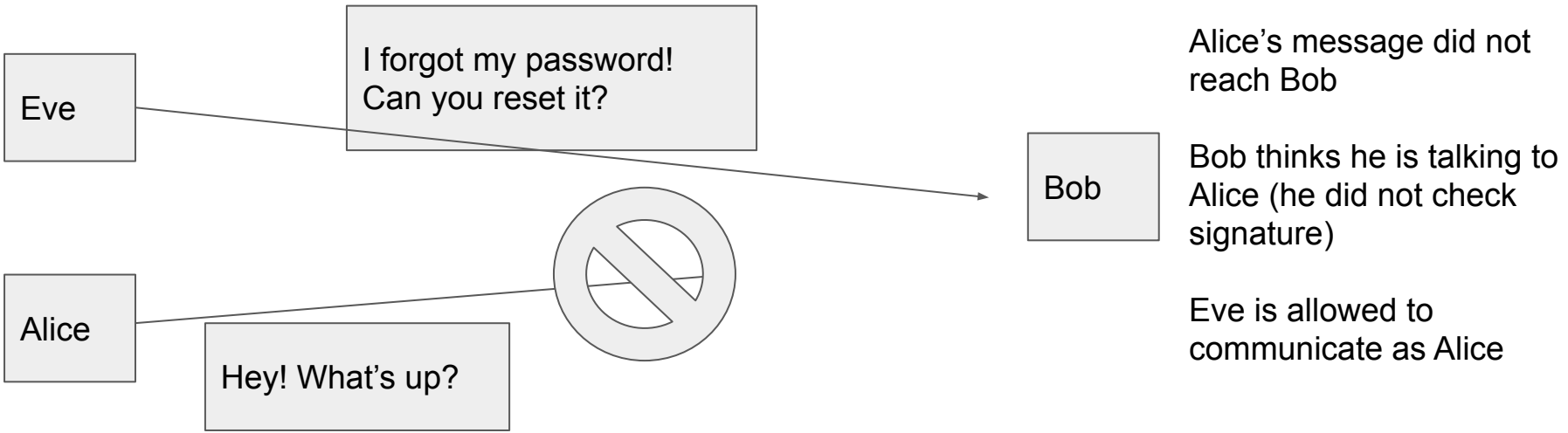
- Why are signatures important in the real world?
- They work similarly here
  - Check file creator

- Sign using YOUR private key
- In this diagram, Alice is sending unencrypted data



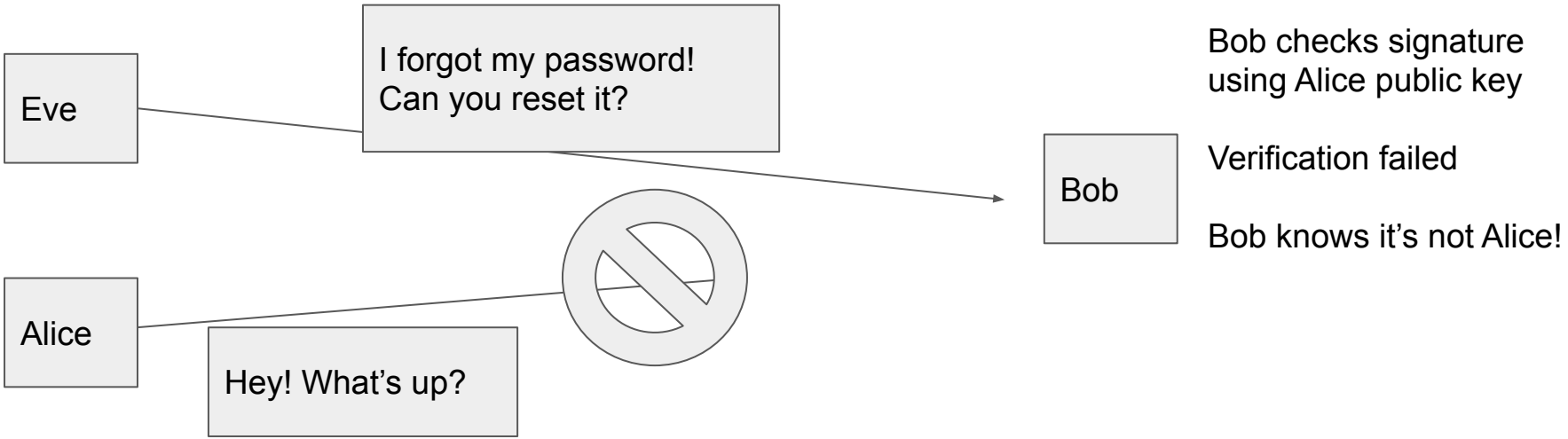
Why do we sign?

- Man in the middle attacks
- We can assume Eve is able to send, intercept, and deny communication



With signatures

- Bob can verify that the message is from Alice



Sign + encrypt

- Alice signs her message, sig = Sign(plaintext, Alice private key)
- Alice encrypts, ciphertext = Encrypt(plaintext + sig, Bob public key)
- Bob decrypts, plaintext + sig = Decrypt(ciphertext, Bob private key)
- Bob verifies, Verify(plaintext, sig, Alice public key)
- Note: the Verify function is checking “plaintext==Decrypt(sig, Alice public key)”

A slight improvement

- Alice hashes her message, hash = F(plaintext)
- Alice creates sig = Sign(hash, Alice private key)
- Alice encrypts, ciphertext = Encrypt(plaintext + sig, Bob public key)
- Bob decrypts, plaintext + sig = Decrypt(ciphertext, Bob private key)
- Bob hashes the message, hash = F(plaintext)
- Bob verifies, Verify(hash, sig, Alice public key)
- Why?
- Is this procedure secure? What else can go wrong?
  - We can assume Eve is able to send, intercept, and deny communication

Who’s public key?

- When Bob tries to get Alice’s public key, Eve intercepts
- Eve instead tells Bob her own public key
- Now Eve pretends to be Alice (she can sign with her own private key!)
- Bob will think it’s Alice
- So how does Bob check the identity of the public key?
- “**Certification authority (CA)** is an entity that issues digital certificates. A digital certificate certifies the ownership of a public key by the named subject of the certificate.” - wikipedia
- We trust the CA!

Questions?

- It’s a little confusing the first time
- Try to go over + understand the need for
  - 1. Public key / private key
  - 2. Signatures
  - 3. CA
- Questions on assignment 2?