* Logistic: quiz on Wednesday

Last Time: Security

* Security Properties: Integrity, Confidentiality, Authenticity
  + Authenticity is necessary for Confidentiality

| Threat Model | Mitigations / Techniques |
| --- | --- |
| Accidental corruption | * checksum/CRC |
| Adversarial modification | * Secure hash * Message Authentication Code (keyed hash) |
| Replay | Idempotence of messages |
| Eavesdropping | encryption (AE AD) |
|  | Authenticated encryption requires a pre-established shared secret. For communication with strangers:   * Trusted Third Party (Kerberos/Windows Active Directory): either relay the connection or Trent generates a new secret key and gives that to Alice and Bob * AKE |

* peers: Alice + Bob
* eavesdropper: Eve
* adversarial modification: Mallory
* trusted third party: Trent
* AKE:

1. Alice generates a key pair: (
2. Bob generates a key pair: (
3. Alice and Bob publishes their public keys
4. Alice sends some to Bob, and Bob sends some to Alice
5. Alice gets and Bob gets , and the secret keys that Alice gets and Bob gets are the same.
6. In addition, knowing , , , does not reveal the secret key.

* But how do we know the public key of say Target?
  + Asking directly from Target does not work, since that message may be corrupted.
  + For a small number of entities, there could be a directory of public keys that were shared in a 100% secure way (e.g. an in-person meeting)
  + Or you could ask someone that you trust and you already know his/her public key
* e.g. You are asking Keith for John’s public key
  + , “John’s public key is <x> according to Keith (expiring at time t)”
  + “John’s public key is <x> according to Keith (expiring at time t)”, . Then, this is a “certificate” that Keith verifies John’s public key is <x>.
  + John can store this certificate, and show this to any person that trusts Keith to prove that John is actually John.
* Firefox —------------TCP----------------- Target @ “target.com”
  + Firefox trusts a list of certification authorities (whose public keys are programmed into Firefox)
  + When Firefox connects to “target.com”, Target, to prove Target is actually Target, would provide:
    - “Hi, I’m target.com. My public key is <x>. Here is a certificate from a CA you trust”.
    - And a certificate: “Target come’s public key is <x> according to <CA>” + from .
  + Firefox:
    - , “Target come’s public key is <x> according to <CA>”,
  + Then Firefox and Target does to get a shared secret key. This shared secret key is used to do for all following communication in the current TCP connection.
  + These steps happen as part of the TLS layer. TLS translates between plaintext and ciphertext
* Q & A
  + A: This list of CAs is common across different browsers.
  + Q: How does a CA decide to give the certificate to a specific entity?  
    A: CA would have an intensive verification process (back in the days), but over the time the standard has been lowered. Now it’s done via domain verification: if someone can put a provided verify.txt at URL/verify.txt within 5 minutes, a CA gives the certificate. This is indeed not secure, since DNS and routing are not secure.
  + Q: What if CAs are forced to grant a certificate?  
    A: Certificate Transparency Log: a log of all certificates granted by CAs. Big companies monitor this
* The shift from HTTP and HTTPS was triggered by the fact governments were monitoring all the traffics (refer to the slides for more information).