# Cybersecurity for Future Presidents

Lecture 10:
DEBATE #1:

Debate 1: Resolved: The U.S. government should mandate that communication and storage technology providers include a mechanism by which protected data can be obtained under lawful court order.

## Any Questions?

- About previous lecture?
- About homework?
- About reading?

My office hours:
Wed. afternoon, 12-3pm, 442 RH

Homework for next week: Reading, Exercises

Reading for next week (for all):

Exercises: Cryptography and applications

## Cybersecurity events from the past week of interest to future (or current) Presidents:

- ✓ Hospital taken offline for a week by ransomware; $3.6M ransom (9,000BTC)
- ✓ http://www.csoonline.com/article/3033160/security/ransomware-takes-hollywood-hospital-offline-36m-demanded-by-attackers.html
- ✓ IRS reports 100,000 eFile credentials compromised, PIN guessing
  - ✓ identity thieves used 464,000 SSNs in unauthorized attempts to access an e-file PIN and were successful in obtaining a PIN in 101,000 of those attempts
  - ✓ https://www.irs.gov/uac/Newsroom/IRS-Statement-on-Efiling-PIN
- ✓ DoJ, HSD employee information published, probably social engineering

Coming up: … ?

## Today's Debate Topic

Debate 1: Resolved: The U.S. government should mandate that communication and storage technology providers include a mechanism by which protected data can be obtained under lawful court order.

## Cryptography basics, continued

## Key Cryptographic Concepts for Future Presidents

- True random numbers vs. pseudo-random numbers
- Perfect Secrecy, and why it's rarely used
- Symmetric cryptography
- Asymmetric (public key) cryptography
  - "trapdoor" or "one-way" functions
- Digital signatures
- Significance of length of key
- Man-in-the-middle attacks

## Random vs. Pseudo-random numbers

(True) Random numbers – generated by physical phenomena, unpredictable, not repeatable (except if you record and replay)

- Flip a coin, toss a die
- Atmospheric noise: see www.random.org
- Radioactive decay
- Radio noise
- Intel on-chip random number generator:
  - thermal noise triggers metastable circuit, output filtered/tested

- Avoid / detect bias: run statistical tests on output
- Looking for a uniform distribution (all outcomes equally likely)
  - Transformations can convert uniform to other distributions

Pseudo random numbers

- A string of random numbers that passes statistical tests for randomness, but is generated deterministically
- Computer program with "seed" or "initialization vector" to provide a starting value; eventually, the stream will cycle

## How to achieve "perfect" secrecy

- Perfect secrecy = no matter how much plaintext/ciphertext eavesdropper may have, still can't decipher a new message
- Believe it or not, this is achievable: ("one-time pad")
- Requires
  - Key bits must be truly random (i.e., generated by a natural random process, not a computer program)
  - Key must never be re-used* to encrypt another message
    - 1 bit of key for each bit of message
    - Recipient must have the same key (and must be able to synchronize the key streams)
- Because the key is random, all decryptions are equally likely – so passive eavesdropper can't determine if proposed decipherment is correct or not.
- Also note that an active eavesdropper (one who can manipulate the encrypted bits) can alter the message received (you get secrecy but not integrity)
- See Anderson, Sec. 5.2.2 (p. 132) for more detail

*Search for 'Venona' for an interesting story of how the Russians misused a one-time pad

## Secret Key (Symmetric) Cryptography

- In symmetric cryptography, the same key is used for encryption and decryption – as in the 'XOR' examples we have done.
- In effect, the key is a random number that provides the seed for a cryptographically secure pseudo-random number generator (CSPRNG); the output of that generator is XOR'ed with the data stream as shown above to generate ciphertext
- The recipient of the message uses the same key to seed the same algorithm, XOR's with the received ciphertext and retrieves the plaintext
- "Key" question: how to get the key to the recipient?
  - Pre-distribute
  - Distribute out-of-band (might be paper, CD, memory stick)
- Passive eavesdropper needs to know the algorithm and determine the key to read the message
- Assuming the cryptoalgorithm is strong, then the eavesdropper needs to test alternative keys by "brute force" – try them out
- Key length then determines the strength of the encryption

## Some problems are hard to compute, but easy to check

Can you think of some?

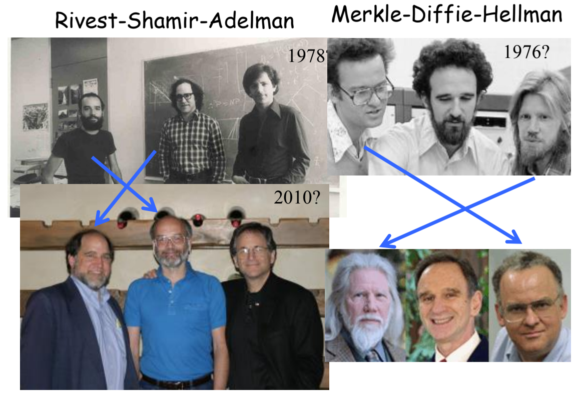
- Finding the square (or cube, or ….) root of a number
- Sudoku
- Finding the prime factors of a large number
- Traveling salesman problem

It turns out that you can use some of these "one-way" or "trapdoor" functions to provide asymmetric or "public key" encryption

## Public Key (Asymmetric) Cryptography

- The sender and the recipient use different keys – one to encrypt and a different one to decrypt (hence asymmetric)
- These schemes rely on the fact that there are "trap-door one-way" functions: functions that are easy to compute in one direction but hard to reverse, unless you know the trap-door
- The most widely used scheme is based on the difficulty of factoring large composite numbers:
  - For two large primes, P and Q, computing N = P*Q is easy
  - But given only N, finding P and Q is hard !
- Rivest-Shamir-Adlemen (RSA) public key encryption uses this fact
- Keys are generated in pairs, [public key, and secret (private) key]
- Plaintext enciphered with one key (public or private) can only be deciphered using the other one
- Each party can make one key public, so that two people who have never communicated privately can, given each others public keys, create a message that can't be read by anyone who doesn't know the private (secret) key
- However, (relative to symmetric crypto algorithms), encryption/decryption are relatively expensive to compute

Rivest-Shamir-Adelman          Merkle-Diffie-Hellman



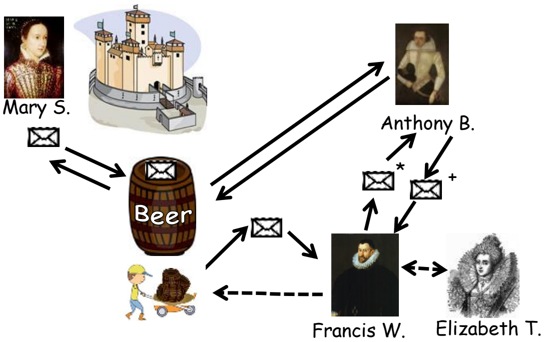
## How Public-Key cryptography is used

- For exchanging a key for a (much faster) symmetric encryption algorithm that will then be used to encrypt communications over a link. (This is what happens in SSL/TLS to secure web communications)
  - Alice picks a symmetric key, encrypts it under Bob's public key and sends to Bob. Bob decrypts it with his private key. They now have a shared symmetric key
  - Issue: how does Alice get the right public key for Bob?
- For signing messages (digital signature):
  - Alice composes message m, then computes "message digest" – a hash of the message, somewhat like a checksum.
  - Alice encrypts the hash with her private key and sends message and hash to Bob
  - Bob receives message with hash; decrypts the hash using Alice's public key; computes the hash of the message and compares with the decrypted hash from Alice – they should match
  - Can be used for both authentication and integrity

## How public key crypto is used on the web

- Public key crypto is a great invention – it seems to solve the key distribution problem. All you need is a phonebook of public keys, right?
  - Yes, but… whose phonebook do you trust?
- Certificate: data structure used to bind an identity to a public key – like the phone book entry
- The phonebook publisher is the Certificate Authority (CA); it has its own public key and signs the phonebook entries using its secret key
- In theory, to get Bob's public key, you communicate with the CA (who may ask a higher level CA, etc.) and get back a certificate with Bob's public key signed by the chain of CA's who endorse it.
- In practice, Bob is likely to be Amazon or Google and Alice is communicating via her browser. The browser comes with a large number of preconfigured Root CA Certificates (I counted over 200 in my store); it will accept connections that are signed by any of those.
- The "Superfish" adware publicized in 2015 abused the certificate system.
- Certificates normally have expiration dates can be revoked if the holder's private key is exposed

## What's a "Man in the Middle" attack, or How Mary Queen of Scots lost her head in 1587




## Cipher used by Mary Queen of Scots and Anthony Babington