Announcements

• Things due
  – Lab 1: tomorrow
  – Homework 2: Next Friday
    • Individual assignment (no groups)
  – CSE 584M: Don’t forget about weekly research readings

• In-class activities
  – 5 “freebies”
  – In-section activities not graded
Stepping Back: Asymmetric Crypto

• Last time we saw session key establishment (Diffie-Hellman)
  – Can then use shared key for symmetric crypto
• Next: public key encryption
  – For confidentiality
• Then: digital signatures
  – For authenticity
Requirements for Public Key Encryption

• Key generation: computationally easy to generate a pair (public key $PK$, private key $SK$)

• Encryption: given plaintext $M$ and public key $PK$, easy to compute ciphertext $C=E_{PK}(M)$

• Decryption: given ciphertext $C=E_{PK}(M)$ and private key $SK$, easy to compute plaintext $M$
  – Infeasible to learn anything about $M$ from $C$ without $SK$
  – Trapdoor function: $\text{Decrypt}(SK, Encrypt(PK, M)) = M$
Some Number Theory Facts

• Euler totient function $\varphi(n)$ ($n \geq 1$) is the number of integers in the $[1,n]$ interval that are relatively prime to $n$
  – Two numbers are relatively prime if their greatest common divisor (gcd) is 1
  – Easy to compute for primes: $\varphi(p) = p-1$
  – Note that $\varphi(ab) = \varphi(a) \varphi(b)$ if $a$ & $b$ are relatively prime
RSA Cryptosystem [Rivest, Shamir, Adleman 1977]

• Key generation:
  – Generate large primes p, q
    • Say, 2048 bits each (need primality testing, too)
  – Compute $n = pq$ and $\varphi(n) = (p-1)(q-1)$
  – Choose small $e$, relatively prime to $\varphi(n)$
    • Typically, $e=3$ or $e=2^{16}+1=65537$
  – Compute unique $d$ such that $ed \equiv 1 \mod \varphi(n)$
    • Modular inverse: $d \equiv e^{-1} \mod \varphi(n)$
  – Public key = $(e,n)$; private key = $(d,n)$

• Encryption of $m$: $c = m^e \mod n$

• Decryption of $c$: $c^d \mod n = (m^e)^d \mod n = m$

How to compute?
- Extended Euclidean algorithm
- Wolfram Alpha 😊
- Brute force for small values
Why is RSA Secure?

- **RSA problem**: given \( c, n=pq, \) and \( e \) such that \( \gcd(e, \varphi(n))=1 \), find \( m \) such that \( m^e = c \mod n \)
  - In other words, recover \( m \) from ciphertext \( c \) and public key \((n,e)\) by taking \( e^{th} \) root of \( c \) modulo \( n \)
  - There is no known efficient algorithm for doing this without knowing \( p \) and \( q \)

- **Factoring problem**: given positive integer \( n \), find primes \( p_1, \ldots, p_k \) such that
  \[
  n = p_1^{e_1} p_2^{e_2} \cdots p_k^{e_k}
  \]

- If factoring is easy, then RSA problem is easy
  (knowing factors means you can compute \( d = \text{inverse of } e \mod (p-1)(q-1) \))
  - It may be possible to break RSA without factoring \( n \) – but if it is, we don’t know how
RSA Encryption Caveats

• Encrypted message needs to be interpreted as an integer less than $n$
• Don’t use RSA **directly** for privacy – output is deterministic! Need to pre-process input somehow.
• Plain RSA also does **not** provide integrity
  – Can tamper with encrypted messages

In practice, OAEP is used: instead of encrypting $M$, encrypt $M \oplus G(r) \parallel r \oplus H(M \oplus G(r))$
  – $r$ is random and fresh, $G$ and $H$ are hash functions
Stepping Back: Asymmetric Crypto

• Last time we saw **session key establishment** (Diffie-Hellman)
  – Can then use shared key for symmetric crypto

• We just saw: **public key encryption**
  – For confidentiality

• Finally, now: **digital signatures**
  – For authenticity
Digital Signatures: Basic Idea

**Given:** Everybody knows Bob’s public key.
Only Bob knows the corresponding private key.

**Goal:** Bob sends a “digitally signed” message
1. To compute a signature, must know the private key
2. To verify a signature, only the public key is needed
RSA Signatures

- Public key is \((n,e)\), private key is \((n,d)\)
- To sign message \(m\): \(s = m^d \mod n\)
  - Signing & decryption are same underlying operation in RSA
  - It’s infeasible to compute \(s\) on \(m\) if you don’t know \(d\)
- To verify signature \(s\) on message \(m\): verify that \(s^e \mod n = (m^d)^e \mod n = m\)
  - Just like encryption (for RSA primitive)
  - Anyone who knows \(n\) and \(e\) (public key) can verify signatures produced with \(d\) (private key)

- In practice, also need padding & hashing
  - Without padding and hashing: Consider multiplying two signatures together
  - Standard padding/hashing schemes exist for RSA signatures
DSS Signatures

• Digital Signature Standard (DSS)
• Public key: \((p, q, g, y=g^x \mod p)\), private key: \(x\)
• Security of DSS requires hardness of discrete log
  – If could solve discrete logarithm problem, would extract \(x\) (private key) from \(g^x \mod p\) (public key)
• Again: We’ve discussed discrete logs modulo integers; significant advantages to using \textit{elliptic curve groups} instead.
Post-Quantum Cryptography

• If quantum computers become a reality
  – It becomes much more efficient to break conventional asymmetric encryption schemes (e.g., factoring becomes “easy”)
  – For block ciphers (symmetric encryption), use 128-bit keys for 256-bits of security

• There exists efforts to make quantum-resilient asymmetric encryption schemes
Cryptography Summary

• **Goal: Privacy**
  – Symmetric keys:
    • One-time pad, Stream ciphers
    • Block ciphers (e.g., DES, AES) → modes: EBC, CBC, CTR
  – Public key crypto (e.g., Diffie-Hellman, RSA)

• **Goal: Integrity**
  – MACs, often using hash functions (e.g., SHA-256)

• **Goal: Privacy and Integrity** (“authenticated encryption”)
  – Encrypt-then-MAC

• **Goal: Authenticity (and Integrity)**
  – Digital signatures (e.g., RSA, DSS)
Want More Crypto?

• Some suggestions:
  – CSE 490C, likely becoming CSE 426: Cryptography
  – Stanford Coursera (Dan Boneh): https://www.coursera.org/learn/crypto
Problem: How does Alice know that the public key she received is really Bob’s public key?
Threat: Person-in-the Middle
Distribution of Public Keys

• Public announcement or public directory
  – Risks: forgery and tampering

• Public-key certificate
  – Signed statement specifying the key and identity
  • $\text{sig}_{CA}(\text{“Bob”}, \text{PK}_B)$

• Common approach: certificate authority (CA)
  – Single agency responsible for certifying public keys
  – After generating a private/public key pair, user proves his identity and knowledge of the private key to obtain CA’s certificate for the public key (offline)
  – Every computer is pre-configured with CA’s public key
You encounter this every day...

SSL/TLS: Encryption & authentication for connections