Public Key Cryptography
Public key exchange: Diffie-Hellmann
What is a cryptosystem?

- $K = \{0,1\}^l$
- $P = \{0,1\}^m$
- $C' = \{0,1\}^n$, $C \subseteq C'$

- $E: P \times K \rightarrow C$
- $D: C \times K \rightarrow P$

- $\forall p \in P, k \in K: D(E(p,k),k) = p$
  - It is \textit{infeasible} to find inversion $F: P \times C \rightarrow K$

Lets start again!
This time in English ... .
What is a cryptosystem?

- A pair of algorithms that take a key and convert plaintexts to ciphertexts and backwards later
  - Plaintext: text to be protected
  - Ciphertext: should appear like random
- Requires sophisticated math!
  - Do not try to design your own algorithms!
The language of cryptography

- **Symmetric or secret key crypto:** sender and receiver keys are identical and secret

- **Asymmetric or Public-key crypto:** encrypt key public, decrypt key secret
Private-Key Cryptography

- traditional private/secret/single key cryptography uses one key
- shared by both sender and receiver
- if this key is disclosed communications are compromised
- also is symmetric, parties are equal
- hence does not protect sender from receiver forging a message & claiming is sent by sender
Public-Key Cryptography

- probably most significant advance in the 3000 year history of cryptography
- uses two keys –
  - a public & a private key
- asymmetric since parties are not equal
- uses clever application of number theoretic concepts to function
- complements rather than replaces private key crypto
Why Public-Key Cryptography?

- developed to address two key issues:
  - **key distribution** – how to have secure communications in general without having to trust a KDC with your key
  - **digital signatures** – how to verify a message comes intact from the claimed sender

- public invention due to Whitfield Diffie & Martin Hellman at Stanford Uni in 1976
  - known earlier in classified community
Public-Key Cryptography

- **public-key/two-key/asymmetric** cryptography involves the use of **two** keys:
  - A **public-key**, which may be known by anybody, and can be used to **encrypt messages**, and **verify signatures**
  - A **private-key**, known only to the recipient, used to **decrypt messages**, and **sign** (create) **signatures**

- is **asymmetric** because
  - Those who encrypt messages or verify signatures **cannot** decrypt messages or create signatures
Public key cryptography
Public-Key Characteristics

- Public-Key algorithms rely on two keys where:
  - it is computationally infeasible to find decryption key knowing only algorithm & encryption key
  - it is computationally easy to en/decrypt messages when the relevant (en/decrypt) key is known
  - either of the two related keys can be used for encryption, with the other used for decryption (for some algorithms)
Public-Key Applications

- can classify use into 3 categories:
  - encryption/decryption (provide secrecy)
  - digital signatures (provide authentication)
  - key exchange (of session keys)

- some algorithms are suitable for all uses, others are specific to one
Security of Public Key Schemes

- like private key schemes brute force exhaustive search attack is always theoretically possible
- but keys used are too large (>512bits)
- security relies on a large enough difference in difficulty between easy (en/decrypt) and hard (crypt-analyse) problems
- more generally the hard problem is known, but is made hard enough to be impractical to break
- requires the use of very large numbers
- hence is slow compared to private key schemes
Public Key Cryptography

*Symmetric key crypto*
- Requires sender, receiver to know shared secret key
- Q: how to agree on key in first place (particularly if never “met”)?
- Q: what if key is stolen?
- Q: what if you run out of keys?
- Q: what if A doesn’t know she wants to talk to B?

*Public key cryptography*
- Radically different approach [Diffie-Hellman76, RSA78]
- Sender, receiver do *not* share secret key
- Encryption key *public*
  (known to *all*)
- Decryption key private
  (known only to receiver)
- Allows parties to communicate without prearrangement
Prime Numbers

- prime numbers only have divisors of 1 and self
  - they cannot be written as a product of other numbers
  - note: 1 is prime, but is generally not of interest
- eg. 2, 3, 5, 7 are prime, 4, 6, 8, 9, 10 are not
- prime numbers are central to number theory
- list of prime number less than 200 is:

  2  3  5  7  11  13  17  19  23  29  31  37  41  43  47  53  59  61
  67  71  73  79  83  89  97 101 103 107 109 113 127 131
  137 139 149 151 157 163 167 173 179 181 191 193
  197 199
Relatively Prime Numbers & GCD

- Two numbers $a$, $b$ are relatively prime if they have no common divisors apart from 1.
  - Eg. 8 & 15 are relatively prime since factors of 8 are 1, 2, 4, 8 and of 15 are 1, 3, 5, 15 and 1 is the only common factor.

- Conversely, can determine the greatest common divisor by comparing their prime factorizations and using least powers.
  - Eg. $300 = 2^1 \times 3^1 \times 5^2$  $18 = 2^1 \times 3^2$ hence $\text{GCD}(18, 300) = 2^1 \times 3^1 \times 5^0 = 6$
Fermat's Theorem

- $a^{p-1} = 1 \pmod{p}$
  - where $p$ is prime and $\gcd(a, p) = 1$

- also known as Fermat’s Little Theorem

- also $a^p = a \pmod{p}$

- useful in public key and primality testing
**Euler Totient Function $\phi(n)$**

- when doing arithmetic modulo $n$
- **complete set of residues** is: $0..n-1$
- **reduced set of residues** is those numbers (residues) which are relatively prime to $n$
  - eg for $n=10$,
  - **complete set of residues** is $\{0,1,2,3,4,5,6,7,8,9\}$
  - **reduced set of residues** is $\{1,3,7,9\}$
- number of elements in reduced set of residues is called the **Euler Totient Function $\phi(n)$**
Euler Totient Function \( \phi(n) \)

- To compute \( \phi(n) \) need to count number of residues to be excluded.
- In general need prime factorization, but
  - for p (p prime) \( \phi(p) = p-1 \)
  - for p.q (p,q prime) \( \phi(pq) = (p-1)(q-1) \)

- Eg.
  - \( \phi(37) = 36 \)
  - \( \phi(21) = (3-1)(7-1) = 2 \times 6 = 12 \)
Euler's Theorem

- A generalisation of Fermat's Theorem
- \( a^{\varphi(n)} = 1 \pmod{n} \)
  - For any \( a, n \) where \( \gcd(a, n) = 1 \)

- Eg.
  - \( a = 3; n = 10; \varphi(10) = 4; \)
    - Hence \( 3^4 = 81 = 1 \pmod{10} \)
  - \( a = 2; n = 11; \varphi(11) = 10; \)
    - Hence \( 2^{10} = 1024 = 1 \pmod{11} \)
Primitive Roots

- from Euler’s theorem have $a^{\phi(n)} \mod n = 1$
- consider $a^m = 1 \pmod{n}$, $\gcd(a, n) = 1$
  - must exist for $m = \phi(n)$ but may be smaller
  - once powers reach $m$, cycle will repeat
- if smallest is $m = \phi(n)$ then $a$ is called a primitive root or generating element
- if $p$ is prime, then successive powers of $a$ "generate" the group $\mod p$
- these are useful but relatively hard to find
the inverse problem to exponentiation is to find the **discrete logarithm** of a number modulo p

that is to find \( x \) such that \( y = g^x \pmod{p} \)

this is written as \( x = \log_g y \pmod{p} \)

if \( g \) is a primitive root then it always exists, otherwise it may not, eg.

\[ x = \log_3 4 \pmod{13} \] has no answer

\[ x = \log_2 3 \pmod{13} = 4 \] by trying successive powers

whilst exponentiation is relatively easy, finding discrete logarithms is generally a **hard** problem
Public-Key distribution of Secret Keys

- use previous methods to obtain public-key
- can use for secrecy or authentication
- but public-key algorithms are slow
- so usually want to use private-key encryption to protect message contents
- hence need a session key
- have several alternatives for negotiating a suitable session
Diffie-Hellman Key Exchange

- first public-key type scheme proposed
- by Diffie & Hellman in 1976 along with the exposition of public key concepts
  - note: now known that Williamson (UK CESG) secretly proposed the concept in 1970
- is a practical method for public exchange of a secret key
- used in a number of commercial products
Diffie-Hellman Key Exchange

- a public-key distribution scheme
  - cannot be used to exchange an arbitrary message
  - rather it can establish a common key
  - known only to the two participants

- value of key depends on the participants (and their private and public key information)

- based on exponentiation in a finite (Galois) field (modulo a prime or a polynomial) – seems easy at first sight

- security relies on the difficulty of computing discrete logarithms (similar to factoring) – hard
Diffie-Hellman Setup

- all users agree on global parameters:
  - large prime integer \( p \)
  - \( a \) being a primitive root mod \( p \)

- each user (eg. A) generates their key
  - chooses a secret key (number): \( x_A < p \)
  - compute their **public key**: \( Y_A = a^{x_A} \mod p \)

- each user makes public that key \( Y_A \)
Diffie-Hellman Key Exchange

User A

- Generate random $X_A < q$
- Calculate $Y_A = \alpha^{X_A} \mod q$
- Calculate $K = (Y_B)^{X_A} \mod q$

User B

- Generate random $X_B < q$
- Calculate $Y_B = \alpha^{X_B} \mod q$
- Calculate $K = (Y_A)^{X_B} \mod q$
Diffie-Hellman Key Exchange

- shared session key for users A & B is $K_{AB}$:
  \[
  K_{AB} = a^{x_A x_B} \mod q
  = y_A^{x_B} \mod q \quad \text{(which B can compute)}
  = y_B^{x_A} \mod q \quad \text{(which A can compute)}
  \]

- $K_{AB}$ is used as session key in private-key encryption scheme between Alice and Bob

- if Alice and Bob subsequently communicate, they will have the same key as before, unless they choose new public-keys

- attacker needs an $x$, thus must solve discrete log, logarithm modulo $q$, i.e., compute $x_A$ from $y_A = a^{x_A}$
Diffie-Hellman Example

- users Alice & Bob who wish to swap keys:

- agree on prime $q=353$ and $a=3$

- select random secret keys:
  - A chooses $x_A=97$, B chooses $x_B=233$

- compute respective public keys:
  - $y_A = 3^{97} \mod 353 = 40$ (Alice)
  - $y_B = 3^{233} \mod 353 = 248$ (Bob)

- compute shared session key as:
  - $K_{AB} = y_B^{x_A} \mod 353 = 248^{97} = 160$ (Alice)
  - $K_{AB} = y_A^{x_B} \mod 353 = 40^{233} = 160$ (Bob)
Key Exchange Protocols

- users could create random private/public D-H keys each time they communicate
- users could create a known private/public D-H key and publish in a directory, then consulted and used to securely communicate with them
- both of these are vulnerable to a meet-in-the-Middle Attack
- authentication of the keys is needed
  - Next lectures more on this!
Summary

- Have considered:
  - Principle of Public Key Cryptography
  - Number Theory basics
  - Diffie-Hellmann Key exchange