# Lecture 7:
# Key Distribution

The era of "electronic mail" [Potter1977] may soon be upon us; we must ensure that two important properties of the current "paper mail" system are preserved: (a) messages are *private*, and (b) messages can be *signed*.

R. Rivest, A. Shamir and L. Adleman. *A Method for Obtaining Digital Signatures and Public-Key Cryptosystems.* Communications of the ACM, January 1978. (The original RSA paper.)

CS588: Security and Privacy
University of Virginia
Computer Science

David Evans
http://www.cs.virginia.edu/~evans

## Traditional Cryptology

- Given a secure channel to transmit a shared secret key, symmetric cryptosystems amplify and time-shift that channel:
  - Can transmit bigger secrets over an insecure channel (except one-time pad)
  - Can transmit later secrets over an insecure channel
- But, the initial secure channel is required

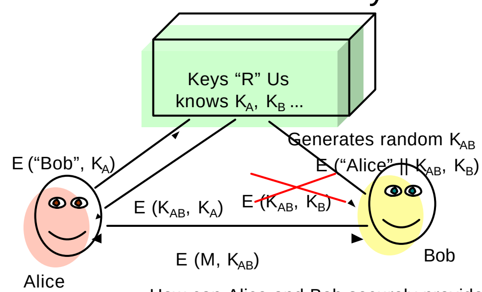


## Key Distribution

- All the cryptosystems we have seen depend on two parties having a shared secret
- Distributing secret keys is hard and expensive
- Can two people communicate securely without having to meet first and establish a key?



## Trust a Third Party




How can Alice and Bob securely provide their keys to Keys "R" Us?



## Merkle's Puzzles

- Ralph Merkle [1974]
- Alice generates $2^{20}$ messages: "This is puzzle $x$. The secret is $y$." ($x$ and $y$ are random numbers)
- Encrypts each message using symmetric cipher with a different key.
- Sends all encrypted messages to Bob



## Merkle's Puzzles, cont.

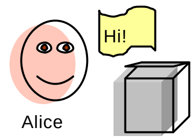
- Bob chooses random message, performs brute-force attack to recover plaintext and secret $y$
- Bob sends $x$ (clear) to Alice
- Alice and Bob use $y$ to encrypt messages

## Is this secure?

- Alice: symmetric cipher DES
  - ~$2^{55}$ expected brute force work to break DES
- Eve: has to break the $2^{20}$ to find which one matches *x*.
  - ~ $2^{19} * 2^{55}$ expected work
- Alice and Bob change keys frequently enough since it is less work to agree to a new key
- Why not increase number of puzzle messages?

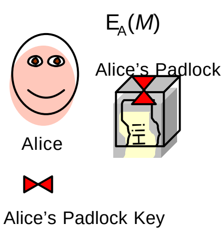


## Padlocked Boxes



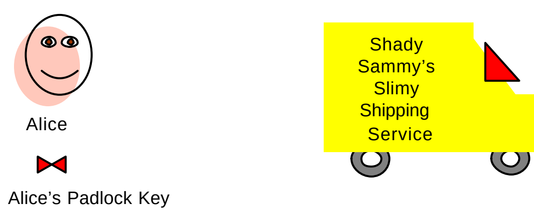



## Padlocked Boxes



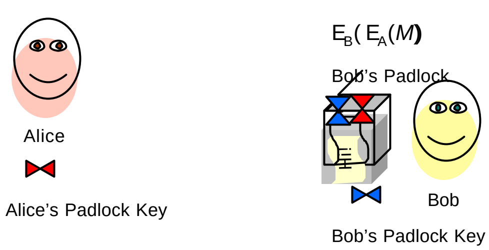



## Padlocked Boxes



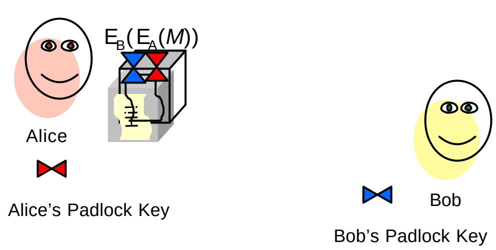



## Padlocked Boxes

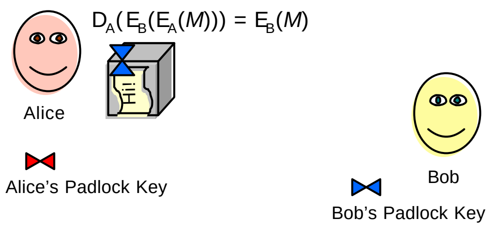





## Padlocked Boxes

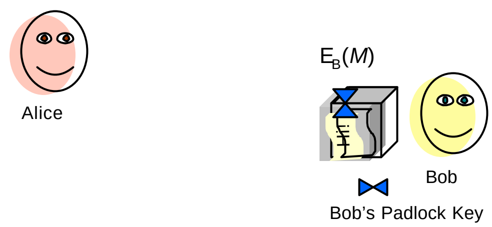


## Padlocked Boxes



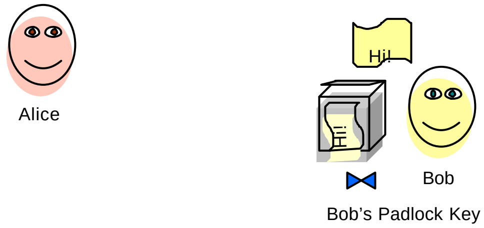



## Padlocked Boxes





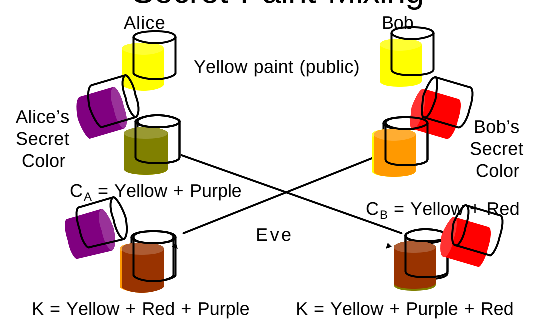

## Padlocked Boxes






Analogy due to Simon Singh, *The Code Book*.

## Secret Paint Mixing






## Birth of Public Key Cryptosystems

- 1969 – ARPANet born: 4 sites
  - Whitfield Diffie starts thinking about strangers sending messages securely
- 1974 – Whitfield Diffie gives talk at IBM lab
  - Audience member mentions that Matrin Hellman (Stanford prof) had spoke about key distribution
- That night – Diffie starts driving 5000km to Palo Alto
- Diffie, Hellman and Ralph Merkle work on key distribution problem



**We stand today on the brink of a revolution in cryptography.** The development of cheap digital hardware has freed it from the design limitations of mechanical computing and brought the cost of high grade cryptographic devices down to where they can be used in such commercial applications as remote cash dispensers and computer terminals. In turn, such applications create a need for new types of cryptographic systems which minimize the necessity of secure key distribution channels and supply the equivalent of a written signature. At the same time, theoretical developments in information theory and computer science show promise of providing provably secure cryptosystems, changing this ancient art into a science. Diffie and Hellman, November 1976.

## Diffie-Hellman Key Agreement

1. Choose public numbers: $q$ (large prime number), $\alpha$ (primitive root of $q$)
2. A generates random $X_A$ and sends B:
   $Y_A = \alpha^{X_A} \bmod q$.
3. B generates random $X_B$ and sends A:
   $Y_B = \alpha^{X_B} \bmod q$.
4. A calculates secret key: $K = (Y_B)^{X_A} \bmod q$.
5. B calculates secret key: $K = (Y_A)^{X_B} \bmod q$.



## What's a primitive root?

- $\alpha$ is a primitive root of $q$ if for all $1 \le n < q$, there is some $m$, $1 \le m < q$ such that
  $\alpha^m = n \bmod q$
- Given $\alpha$, $n$ and $q$ can we solve for $m$?
  - Yes: there is only one possible $m$
  - But, it might be hard to find
- Discrete logarithm: given $\alpha$, $n$, and $q$ find $0 \le m < q$ such that $\alpha^m = n \bmod q$.



## Example

- What is a primitive root for $q = 11$?

| | |
|---|---|
| $2^1 \equiv_{11} 2$ | $2^6 = 64 \equiv_{11} 9$ |
| $2^2 \equiv_{11} 4$ | $2^7 = 128 \equiv_{11} 7$ |
| $2^3 \equiv_{11} 8$ | $2^8 = 256 \equiv_{11} 3$ |
| $2^4 = 16 \equiv_{11} 5$ | $2^9 = 512 \equiv_{11} 6$ |
| $2^5 = 32 \equiv_{11} 10$ | $2^{10} = 1024 \equiv_{11} 1$ |



## Finding Primitive Roots

- Theorem: All prime numbers have primitive roots.
  - Book proves this using Proof by Forward Reference
    "*(Proof later.)*" (p .137) and "*this will be proven later*" (p. 230), "*which will be proven only later*" (p. 231), "*which is known to exist*" (p. 445).
  - We'll use the same technique
  - In practice, it is easy to find primitive roots for prime numbers by guessing. Almost ½ of guesses will work (next class we will see why).



## Diffie-Hellman Example

1. Choose public numbers: $q$ (large prime number), $\alpha$ (generator mod $q$):
   $q = 11, \alpha = 2$
2. A generates random $X_A$ and sends B:
   $Y_A = \alpha^{X_A} \bmod q$.
   $X_A = 4$, $Y_A = 2^4 \bmod 11 = 16 \bmod 11 = 5$
3. B generates random $X_B$ and sends A:
   $Y_B = \alpha^{X_B} \bmod q$.
   $X_B = 6$, $Y_B = 2^6 \bmod 11 = 64 \bmod 11 = 9$

Example from Tom Dunigan's notes: http://www.cs.utk.edu/~dunigan/cs594-cns00/class14.html



## Diffie-Hellman Example, cont.

$q = 11, \alpha = 2$
$X_A = 4, Y_A = 5 \quad X_B = 6, Y_B = 9$

4. A calculates secret key: $K = (Y_B)^{X_A} \bmod q$.
   $K = 9^4 \bmod 11 = 6561 \bmod 11 = 5$.
5. B calculates secret key: $K = (Y_A)^{X_B} \bmod q$.
   $K = 5^6 \bmod 11 = 15625 \bmod 11 = 5$.

## Is it magic? Things to Prove:

1. They generate the same keys:
   $K = (Y_B)^{X_A} \bmod q = (Y_A)^{X_B} \bmod q$
2. An eavesdropper cannot find $K$ from any transmitted value:
   $q$, $\alpha$, $Y_A$, $Y_B$



## 1. Keys Agree

- Prove $K = (Y_B)^{X_A} \bmod q = (Y_A)^{X_B} \bmod q$.

| | |
|---|---|
| $(Y_B)^{X_A} \bmod q$ | $(Y_A)^{X_B} \bmod q$ |
| $= (\alpha^{X_B} \bmod q)^{X_A} \bmod q$ | $= (\alpha^{X_A} \bmod q)^{X_B} \bmod q$ |
| $= (\alpha^{X_B})^{X_A} \bmod q$ | $= (\alpha^{X_A})^{X_B} \bmod q$ |
| $= \alpha^{X_B X_A} \bmod q$ | $= \alpha^{X_A X_B} \bmod q$ |

QED.



## Modular Exponentiation

$(a \bmod q)^b \bmod q = a^b \bmod q$

$(7 \bmod 6)^2 \bmod 6 = 7^2 \bmod 6$

$1^2 \bmod 6 = 49 \bmod 6$

Proof by example?



## Modular Exponentiation

- First prove:
  $(a * b) \bmod q = (a \bmod q) * (b \bmod q) \bmod q$
- Then, by induction,
  $(a \bmod q)^b \bmod q = a^b \bmod q$
  since $a^b = a * a^{b-1}$ and $a^1 = a$.



## Modular Arithmetic

$(a * b) \bmod n = x$

$x + (n * d0) = a * b$

$x = a * b - (n * d0)$

$a \bmod n = y \Rightarrow y = a - (n * d1)$

$b \bmod n = z \Rightarrow z = b - (n * d2)$

$(a \bmod n) * (b \bmod n) \bmod n$

$= (a - (n * d1)) * (b - (n * d2)) \bmod n$

$= (a * b + (a * (n * d2)$

$- b * (n * d1) + (n * d1)(n * d2)) \bmod n$

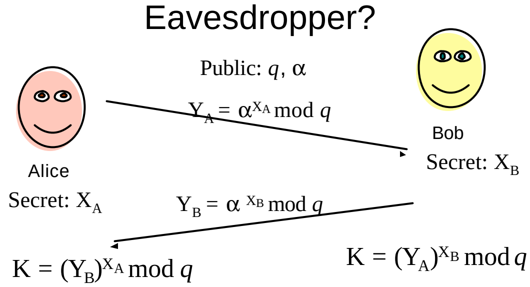
$= a * b \bmod n$ (all terms with $n *$ are $0 \bmod n$)



## 2. Secure from Eavesdropper

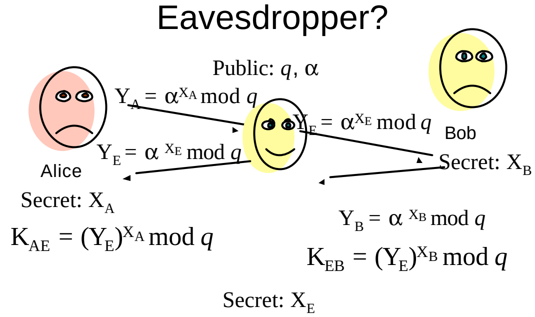
- An eavesdropper cannot find
  $K = (Y_B)^{X_A} \bmod q = (Y_A)^{X_B} \bmod q$
  from any transmitted value:
  $q$, $\alpha$, $Y_A = \alpha^{X_A} \bmod q$, $Y_B = \alpha^{X_B} \bmod q$
- Attacker needs to solve $Y_A = \alpha^{X_A} \bmod q$ for $X_A$
- Finding discrete logarithms is (probably) hard!
  - Best known algorithm: $e^{((\ln q)^{1/3} \ln (\ln q))^{2/3})}$

## Secure from Active Eavesdropper?






## Secure from Active Eavesdropper?






## Diffie-Hellman Use

- SSL
- Cisco encrypting routers
- Sun secure RPC
- etc...



## Public-Key Cryptography

- Same paper introduced concept of Public-Key Cryptography
- Public algorithm: E
- Private algorithm: D
- Identity: $\mathrm{E}(\mathrm{D}(m)) = \mathrm{D}(\mathrm{E}(m)) = m$
- Secure: cannot determine E from D
- But didn't know how to find suitable E and D



## Knapsack Ciphers

- [Merkle, Hellman 78]
- Knapsack Problem:
  - Given positive integers $a_1, a_2, \ldots, a_n$ and a positive integer $b$ find a subset of $a$'s that sum to $b$.
  - In general, this is NP-complete
    - Can try $2^n$ possible subsets, check each one in polynomial time
    - If we could solve it in polynomial time, we could solve all other NP problems in P also
    - Proof: reduce to satisfiability (~ vehement assertion)



## Encryption

- Message = $(x_1, \ldots., x_n)$ (bit vector)
- Knapsack vector: $a = (a_1, \ldots., a_n)$
- Ciphertext: $b = x_1a_1 + x_2a_2 + \ldots + x_na_n$
- Decrypt by finding subset of $a_i$'s that sum to b. Message bits corresponding to $i$'s are 1.
- Unique decryption?
  - Depends on choice of knapsack: can't have duplicate elements, can't have elements equal to sum of subset of other elements

## Superincreasing Knapsack

- $a = (a_1, …., a_n)$ where for all $i$,
  $a_i > a_1 + a_2 + … + a_{i-1}$
- If $a$ is superincreasing, how hard is decryption?

```
for i = n to 1 step –1
    if b >= a_i then
        b_i = 1  b = b – a_i
    else
        b_i = 0
```



## Disguise the Knapsack

Instead of using $a = (a_1, …., a_n)$ use
$c = (ta_1 \bmod m, …., ta_n \bmod m)$
where
$m > a_1 + a_2 + … + a_n$
and $t$ is random secret, relatively prime to $m$
(Hence, there is an inverse $t^{-1} \bmod n$)

Alice publishes $c$ as her "public knapsack".



## Knapsack Encryption

To send a message,
$b = x_1c_1 + x_2c_2 + … + x_nc_n$
Alice decrypts by:
$t^{-1} b \bmod m = t^{-1}x_1c_1 + t^{-1}x_2c_2 + … + t^{-1} x_nc_n$
$c = (ta_1 \bmod m, …., ta_n \bmod m)$ so
$t^{-1}x_1c_1 = a_1x_1 \bmod m$
$t^{-1} b \bmod m = x_1a_1 + x_2a_2 + … + x_na_n$
Easy for Alice to compute $x$'s now using superincreasing knapsack.



## Example

Private key: (3, 5, 9, 20, 44)
t = 67, m = 89
$t^{-1}$ = 4 since 67 * 4 = 1 mod 89
3 * 67 = 201 mod 89 = 23, …
Public key: (23, 68, 69, 5, 11)
Encrypt M = (01011)
C = 68 + 5 + 11 = 84
Decrypt
C * 4 = 69 mod 89
= 5 + 20 + 44 = (01011)



## Knapsack Security?

- Security relied on proof that solving general knapsack problem is NP-hard
- But, adversary doesn't have to solve general knapsack problem – just convert to superincreasing knapsack
- Shamir [1983] showed it is possible to do this in polynomial time without known $t$ and $m$
- Lesson: just because a cipher uses a provably hard problem, doesn't mean there isn't a way of breaking the cipher without solving that problem



## Charge

- Next time:
  - Rivest, Shamir, Adelman: First solution to finding suitable E and D
    - Identity: $E(D(m)) = D(E(m)) = m$
    - Secure: cannot determine E from D
- Read the paper!
  - Go somewhere appropriate: this is perhaps the most important paper in past 30 years!
  - Identify 2 questionable statements in the paper