# PUBLIC KEY ALGORITHMS

Diffie – Hellman
Digital Signature Standard
Zero Knowledge Proof

# DIFFIE - HELLMAN

- Overview
- Method
- Bucket Brigade Attack
- Safe Primes

# Diffie Hellman Overview

- Predates RSA and still in use
- Public exchange yields shared secret
- Vulnerable to impersonation without authentication

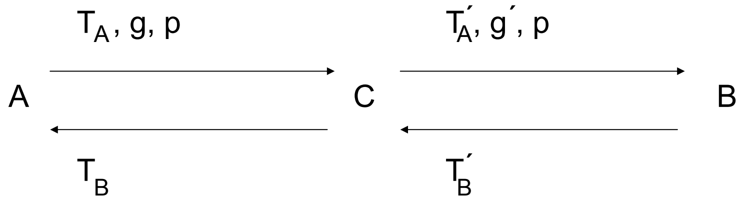
# Diffie – Hellman Method (I)

- Large prime p and g (< p) are made public
- Parties A and B each choose 512 – bit secrets $S_A$ and $S_B$
- A calculates $T_A = g^{S_A} \bmod p$
- B calculates $T_B = g^{S_B} \bmod p$
- A and B exchange $T_A$ and $T_B$

# Diffie – Hellman Method (II)

- A calculates $T_B^{S_A}$, B calculates $T_A^{S_B}$
- A and B have shared secret $g^{S_A S_B}$ mod p
- An attacker needs $S_A$ from $T_A$ or $S_B$ from $T_B$
- Discrete log problem is computationally infeasible

# Bucket Brigade Attack




A and B have no knowledge of C who has arranged a shared secret with both A and B for secure communications

# Defences against bucket brigade attack

- Numbers p and g published securely
- Authenticated Diffie – Hellman

  - A encrypts exchange with B's public key and vice versa

  - A signs exchange with private key

  - Authenticate by hashing Diffie – Hellman exchange with shared secret

# Safe Primes

- If p is prime and

  - (p – 1)/2 is prime, and

  - $g^x \neq_x 1$ mod p unless x = 0 mod p-1, then

  p is a safe prime

  p and g should be changed regularly

# DSS Algorithm

- Generating DSS
- Verifying DSS
- Analysis
- Security strength

# Generating DSS(I)

- choose 160-bit prime q and 512-bit prime $p = kq + 1$
- find g such that $g^q = 1 \bmod p$
- choose long term public/private key pair such that $S < q$ and $T = g^S \bmod p$

# Generating DSS(II)

- choose per message public/private pair $(T_m, S_m)$, $S_m < q$, $T_m = ((g^{S_m} \bmod p) \bmod q)$
- find $MD(m) = d_m$
- Signature $X = S_m^{-1}(d_m + ST_m) \bmod q$
- Transmit $m, T_m, X$

# Verifying DSS

- find $X^{-1} \bmod q$
- find $d_m$
- calculate $x = d_m . X^{-1} \bmod q$
- calculate $y = T_m . X^{-1} \bmod q$
- calculate $z = (g^x T^y \bmod p) \bmod q$
- verified if $z = T_m$

# DSS Analysis

- let $v = (d_m + ST_m)^{-1} \bmod q$
- $X^{-1} = S_m(d_m + ST_m)^{-1} = S_m v \bmod q$
- $x = d_m X^{-1} = d_m S_m v \bmod q$
- $y = T_m S_m v \bmod q$
- $z = g^{d_m S_m v} g^{ST_m S_m v} = g^{S_m} = T_m \bmod p \bmod q$ $(g^q = 1 \bmod p)$

# DSS Security Strength

- Private key S not divulged
- Attacker cannot sign without S
- Attacker cannot find new message to match signature
- Attacker cannot modify message and maintain valid signature

# Zero Knowledge Proof Systems

- Used for authentication
- A proves to B that A has a secret without revealing that secret to B
- Most ZKF systems much faster than RSA
- Can be adapted for signatures

# Square Root Method (I)

- Public key (n, v)
- n = p x q ( p, q large primes)
- v is a number for which secret is $\sqrt{v} \bmod n$
- v is found by choosing random s and calculating $s^2 \bmod n$
- A holds the public key (n, v) and will prove that A knows $\sqrt{v}$

# Square Root Method (II)

- A selects k random numbers $r_1$, $r_2$,..$r_k$ and sends $r_i^2$ to B
- B chooses randomly subsets 1 and 2 of $r_i^2$
- A sends $sr_i$ mod n for 1 and $r_i$ mod n for 2
- B squares values and checks $vr_i^2$mod n and $r_i^2$ mod n
- Valid method because finding square roots mod n is as difficult as factoring n

# Signatures by square root method

- Signature is zero knowledge proof with artificial challenge
- "Challenge" derived from digest of message to be signed
- Concatenate message with k $r_i^2$mod n
- Each bit of MD corresponds to a challenge
- Signature is k values of $r_i$ and k responses to the challenge