Computer and Network Security

Lecture 7

Public Key Cryptography

Facts about PK

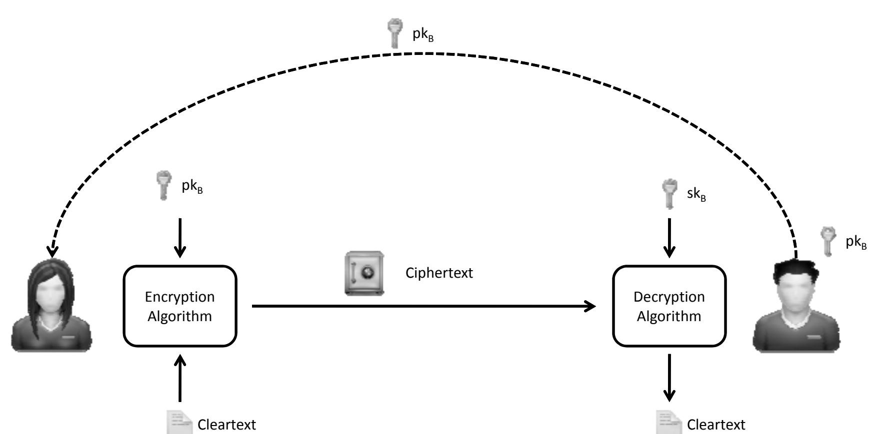
- Euler's totient function
  - $\phi(n)$ = # of positive integers ≤ n AND coprime with n
  - n prime → $\phi(n) = n-1$
- Lagrange's theorem (corollary)
  - If $g \in Z_n^*$ → $g^{\phi(n)} = 1 \bmod n$
- Euclidean algorithm
  - Which is the inverse of x in Z_n ?
- Chinese reminder theorem
  - If n=pq and p,q primes → computation in Z_n can be performed faster in $Z_p \oplus Z_q$

Outline

- Diffie-Hellman Key Exchange
- RSA Encryption
- Digital Signature
  - RSA
- Identification scheme
  - ZeroKnowledge

Public-key (asymmetric) Cryptography

- Bob has a public/private key pair (pk_B, sk_B)
  - Examples: RSA, El Gamal



Diffie-Hellman key exchange

- Protocol to establish a secret key between two parties
- Appeared as
  - New Directions in Cryptography
    - Witfield Diffie and Martin E. Hellman
    - IEEE Transactions on Information Theory (1976)
- System parameters:
  - p large prime
  - g generator of Z^*_p={1, …, p-1}

pick $\mathbf{x_A}$ at random
$y_A = g^{x_A}$ mod p

y_A →

← y_B

pick $\mathbf{x_B}$ at random
$y_B = g^{x_B}$ mod p

$k_{AB} = (y_B)^{x_A}$
$= (g^{x_B})^{x_A}$
$= g^{x_B x_A}$ mod p

$k_{BA} = (y_A)^{x_B}$
$= (g^{x_A})^{x_B}$
$= g^{x_A x_B}$ mod p

Learns g,p, y_A, y_B

Discrete Logarithm Problem (DLP)

- Z^*_n = set of integers mod n, relatively prime to n
- p prime, Z^*_p = {1, …, p-1}, cyclic multiplicative group
  - g generator of Z^*_p
    - Z^*_p = ={g^0 mod p, g^1mod p, …, g^{p-2} mod p}
- Discrete exponentiation is easy
  - Given any x, compute $y = g^x$ mod p
- Discrete logarithm is hard
  - Given any y, find x : $y = g^x$ mod p
    - That if, find $x=\log_g(y)$ mod p

$x_A = \log_g(y_A)$???

Example

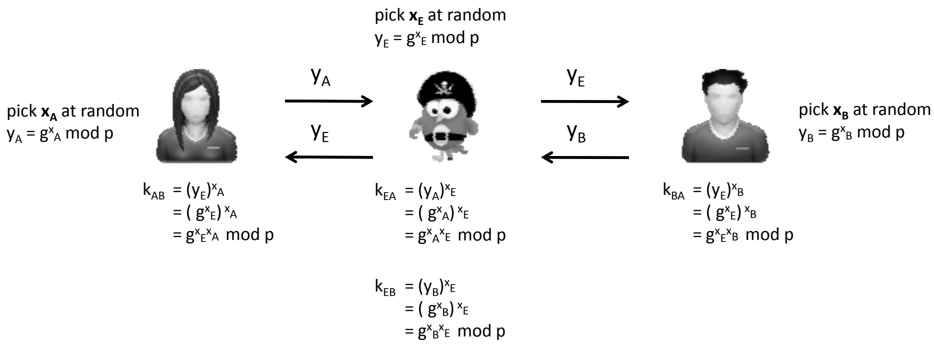
- Discrete exponentiation in Z^*_{17}
  - $3^4 = 81 = 13 \bmod 17$ → $3^4 = 13 \bmod 17$
- Discrete Logarithm in Z^*_{17}
  - $3^x = 13 \bmod 17$, solve for x
  - 4 is a solution
  - 4+16n, for any n, is a solution
    - ord(3) = 16 → $3^{16} = 1 \bmod 17$
    - $3^{4+16n} = 13 * 1^n \equiv 13 \bmod 17$
  - Infinite solutions

Algorithms to find DL

- Trial multiplications
- Baby-step giant-step
- Pollard's rho and lambda algorithms
- Pohlig-Hellman algorithm
- Index calculus algorithm
- Number field sieve
- Function field sieve

None of them runs in polynomial time!

Man-in-the-Middle Attack

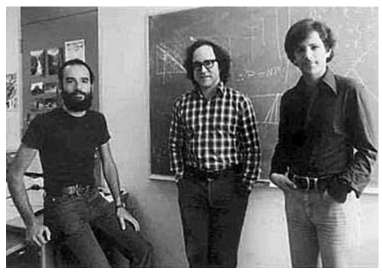


PK Encryption Scheme

- Keypair
  - **sk, pk ∈ *K***
- Plaintext (cleartext)
  - **m ∈ *M***
- Ciphertext
  - **c ∈ *C***
- Encryption algorithm
  - **c = Encrypt(pk, m)**
- Decryption algorithm
  - **m = Decrypt(sk, c)**

The RSA Cryptosystem

- A method for obtaining digital signatures and public-key cryptosystems
  - Ronald L. Rivest, Adi Shamir, Leonard Adleman
  - Communications of the ACM, 1978
- Current use
  - SSL/TLS: Certificates and key-exchange
  - Secure e-mail: PGP, Outlook, …



The RSA trapodoor 1-to-1 function

- Parameters
  - N=pq : N ≈1024 bits, p,q ≈512 bits
- Public Key
  - e : gcd(e, φ(N)) = 1
    - φ(N) = (p-1)(q-1)
- Secret key
  - d : e*d = 1 mod φ(N)
- Encryption of $m \in Z^*_N$
  - $c = E_e(m) = m^e \bmod N$
- Decryption of $c \in Z^*_N$
  - $m = D_d(c) = c^d \bmod N$

The RSA trapodoor 1-to-1 function

- Does it work?
  - $m = D_d(c) = c^d \bmod N = (m^e)^d \bmod N = \underline{m^{ed} \bmod N = m \bmod N}$
  - if $\gcd(ed, \varphi(N)) = 1$ → $ed = 1 \bmod \varphi(N)$
  - $m^{ed} \bmod N = m^{k*\varphi(N)+1} \bmod N$ (Lagrange: $g^{\varphi(N)} = 1 \bmod N$)
  - $m^{k*\varphi(N)+1} \bmod N = m^{k*\varphi(N)} m^1 \bmod N = 1^k m^1 \bmod N = m \bmod N$
- Is it secure?
  - Given (N, e) and a $c = m^e$ **mod** N, it is hard to efficiently compute m
  - Best strategy is to find factor of N (e.g., p, q)
  - Integer factorization seems not practical for large N
    - But there is not proof

Example

- $p = 5, q = 7, n = 35, \varphi(35) = (p-1)(q-1) = 24, e = 11, d = 11$
- m = 2, $c = E_{11}(2) = 2^{11} \bmod 35 = 18 \bmod 35$
- c = 18, $m = D_{11}(18) = 6.426841007923e+13 \bmod 35 = 2$
- $p = 17, q = 13, n = 221, \varphi(221) = (p-1)(q-1) = 192, e = 5, d = 77$
- $m = 5, c = E_5(5) = 5^5 \bmod 221 = 31 \bmod 221$
- $c = 31, m = D_{77}(31) = 6.83676142775442000196395599558e+114 \bmod 221 = 5$

Textbook RSA is insecure

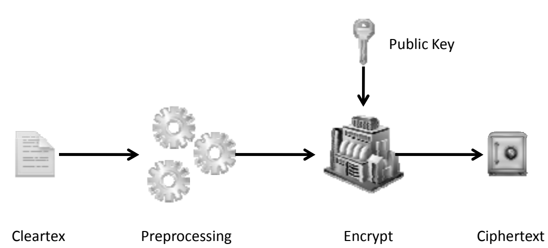
- Textbook RSA encryption:
  - public key: **(N,e)** Encrypt: $\mathbf{c = m^e} \bmod N$
  - private key: **d** Decrypt: $\mathbf{m = c^d} \bmod N$

  $(\mathbf{m \in Z^*_N})$

- Completely insecure cryptosystem:
  - Does not satisfy basic definitions of security
  - Many attacks exist

RSA Encryption in practice

- OAEP+ (Shoup'01)
  - Widely used in web browsers



How to pick your public key

- Pick 2 primes, p and q
- Compute N = pq and $\phi(N) = (p-1)(q-1)$
- Choose a random e ($1 < e < \phi(n)$)
  - $\gcd(e, \phi(n)) = 1$
- Compute $d = e^{-1} \bmod \phi(N)$
  - $ed = 1 \bmod \phi(N)$
  - Extended Euclidean algorithm
- Public Key
  - (e, N)
- Private Key
  - (d,N)

Bob picks its public key

- Random p = 59, q = 67
  - N = 3953
  - $\phi(N) = 58*66 = 3828$
- Random $1 < e < 3828$
  - Let’s try 2669. Will that work? gcd(2669, 3828) = 1
- Now find d, such that $ed = 1 \bmod \phi(n)$
  - d * 2669 = 1 mod 3828
  - d exists ↔ gcd(d, 3828) = 1
  - d = 1625
- pk_B = (2669, 3953)
- sk_B = (1625, 3953)

Message exchange

- Alice
  - pk_B = (2669, 3953)
  - m = 3128
  - $E_{pk_B}(3128) = 3128^{2669} \bmod 3953 = 3541$
- Bob
  - sk_B = (1625, 3953)
  - $D_{pk_B}(3541) = 3541^{1625} \bmod 3953 = 3128$

RSA – Security

- Suppose Eve sees $c = m^e \bmod N$
  - Can she recover m?
- It is strongly believed, but not proven, that Eve cannot efficiently revert c without knowing $\phi(N)$
- It has been proven that finding $\phi(N)$ is equivalent to factoring N
- It is strongly believed, but not proven, that factoring large numbers is difficult

RSA performance dilemma

- Greater security = Longer keys
- Encryption/Decryption time increases cubically with key size
- RSA has poor performance
  - Get worse as algorithms improve and security requirements increase
  - Never used for full communication
    - Used to encrypt a session key

RSA operations

- Finding prime numbers and testing primality
  - Agrawal, Kayal, Saxena (2002)
    - polynomial time
- Exponentiation
  - Square and multiply
- Factorization
  - Believed to be difficult (security is here)

Square and Multiply

- Suppose we want to compute 3^{41} mod 187
  - Can we do better 3 * 3 *3 * … * 3 mod 187?
- Compute squares
  - $3^1 \equiv 3$ mod 187 $3^2 \equiv 9$ mod 187 $3^4 \equiv 81$ mod 187
  - $3^8 \equiv 16$ mod 187 $3^{16} \equiv 69$ mod 187 $3^{32} \equiv 86$ mod 187
- Write 41 in binary
  - 101001 = 32 + 8 + 1.
- Finally, multiply the appropriate powers of two:
  - $3^{41} \equiv 3^1 \cdot 3^8 \cdot 3^{32} \equiv 3 \cdot 16 \cdot 86 \equiv 14 \pmod{187}$
- At most $2 \cdot \log_2(N)$ multiplications

Factorization

- Brute-force
  - For d = 1, 2, 3, 4,…
    - Does d divide n?
  - N = pq
    - If $p \leq q \rightarrow N \geq p^2 \rightarrow \sqrt{N} \geq p$
  - $O(\sqrt{N})$
- Use structure of Z_n
  - Pollard's rho method
  - Quadratic sieve, Number Field Sieve (NFS)
  - Is there a better method out there?

Factoring

- Suppose N (663 bits)
  - 27997833911221327870829467638722601621070446786955428537560009929326128400107609345671052955360856061822351910951365788637105954482006576775098580557613579098734950144178863178946295187237869221823983
  - What are p and q?
- RSA-200 challenge
  - broken 5/9/05
  - Jens Franke’s team at the University of Bonn, Germany
  - Their prize was $20,000 US
- p=3532461934402770121272604978198464368671197400197625023649303468776121253679 423200058547956528088349
- q=7925869954478333033347085841480059687737975857364219960734330341455767872818 152135381409304740185467

General Number Field Sieve

- Complexity $O\left(\exp\left(\left(\frac{64}{9}\log n\right)^{\frac{1}{3}}(\log\log n)^{\frac{2}{3}}\right)\right)$
- So a 663 bit number does not require 2^{663} work to factor.
  - Typically a 1024 bit value requires roughly 2^{80} work to factor
- The U.S. Government uses N values as high as 15360 bits for TOP SECRET communications

Key length and complexity

- **Bits of security** are used to indicate the strength of a cryptographic system
  - < 40 bits — Totally insecure, can be cracked on a PC
  - 56 bits — Key length of DES, 1976 standard (now obsolete)
  - 64 bits — Largest publicly cracked keys, by Distributed.net
  - 70-80 bits — Allegedly searchable by the NSA
  - 128 bits — Standard for AES, as of 2001
  - 256 bits — Required for US Government TOP SECRET material

- Typical RSA: 1024 bit length provides roughly 80 bits of security

Digital Signatures

- Digital equivalent of regular (paper-based) signature
  - Integrity
  - Authentication
  - Non-repudiation
  - Authorization

Application

- Digital Signatures
  - Alice computes signature σ message m
    - Uses her secret key
    - Forgery should not work
  - Anybody can verify (σ, m)
    - Using Alice's public key

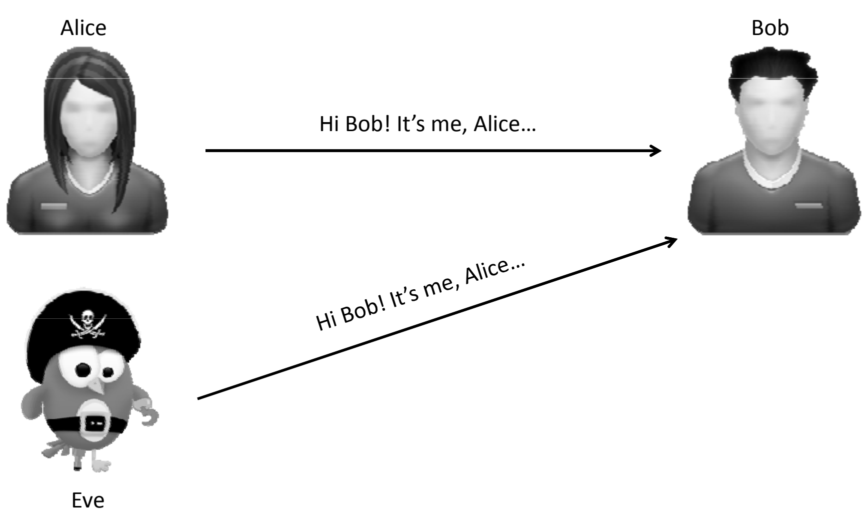
Digital Signature Scheme

- Keypair
  - **sk, pk ∈ *K***
- Plaintext (cleartext)
  - **m ∈ *M***
- Signature
  - **σ ∈ *S***
- Signing algorithm
  - **σ = Sign(sk, m)**
- Verification algorithm
  - **{0,1} = Verify(pk, σ, m)**

RSA Signature scheme

- Parameters
  - N=pq : N ≈1024 bits, p,q ≈512 bits
- Public Key
  - e : gcd(e, φ(N)) = 1
    - $\varphi(N) = (p-1)(q-1)$
- Secret key
  - d : $e*d = 1 \mod \varphi(N)$
- Signing of $m \in Z^*_N$
  - $Sign(d, m) = m^d \mod N$
- Verification of $\sigma \in Z^*_N$
  - Verify(e, σ)
    - Checks if $m = \sigma^e \mod N$

Identification

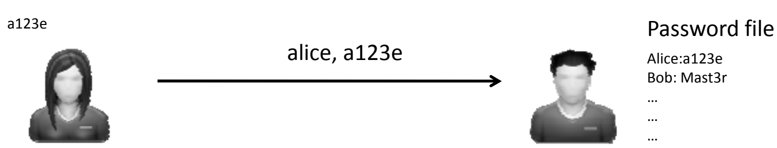


Identification

- (sometimes) Interactive protocol
  - Two parties
    - Prover 
    - Verifier 
    - Alice convinces Bob that she is indeed Alice
- Complete
  - Alice can convince Bob that she indeed is Alice
- Sound
  - Anyone else can convince Bob to be Alice with small probability
- Zero-Knowledge
  - Informally, Alice leaks no information through the protocol

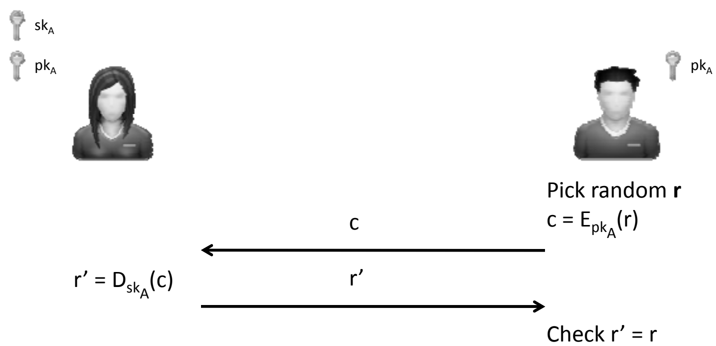
33

First attempt – Password scheme



- Complete and sound
- Non ZK
  - Alice reveals the password

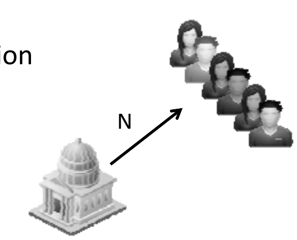
Second attempt – PK based scheme



- Complete and sound
- Non ZK
  - Alice reveals cleartext corresponding to c

Fiat-Shamir Identification Scheme

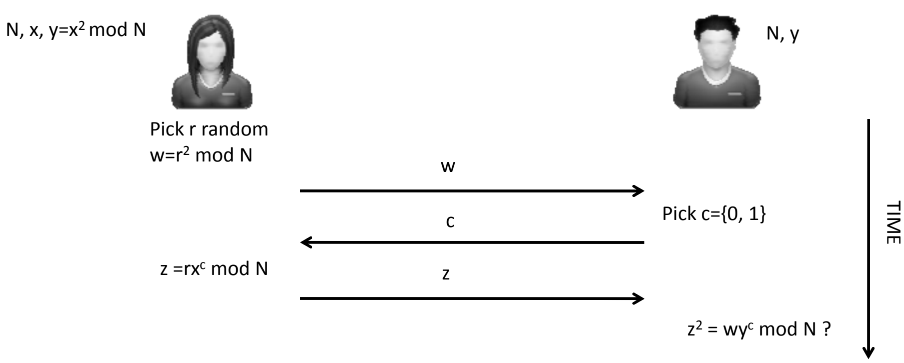
- How To Prove Yourself: Practical Solutions to Identification and Signature Problems
  - Amos Fiat and Adi Shamir
  - CRYPTO 1986
- Based on RSA modulus N=pq
  - Factors themselves are not used in the protocol
  - More provers can share same N
    - As long as nobody know the factorization
    - Trusted center can generate it
      - Delete factors after computation



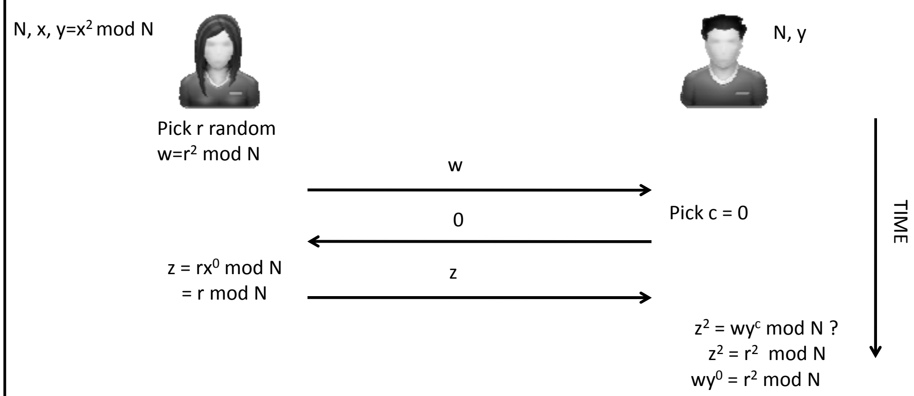
Fiat-Shamir Identification Scheme

- Setup
  - Secret key $1 < x < N$ $\gcd(x,N) = 1$
  - Public key (y, N) $y = x^2 \bmod N$
- Given (y, N), Alice convinces Bob
  - knowledge of **x,** such that, $\mathbf{y = x^2 \bmod N}$
    - that is, knowledge of **a square root of y mod N**
  - Without revealing x

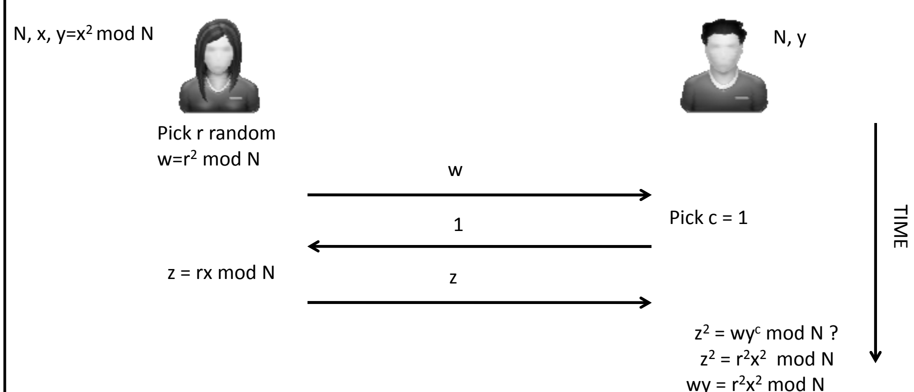
Fiat-Shamir Identification Scheme



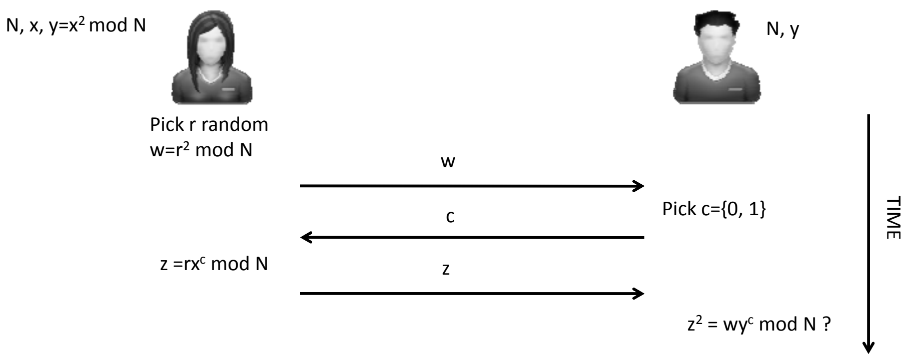
Fiat-Shamir Identification Scheme



Fiat-Shamir Identification Scheme



Fiat-Shamir Identification Scheme



- **z^2** is from secret and challenge
- **wy^c** is from public key, witness and challenge
  - $z^2 = (rx^c)^2 = r^2x^{2c} = wy^c$
- Protocol is repeated many times (e.g., 20, 30, log(N))
  - If Alice is successful in al runs, Bob concludes that he is talking to Alice

Security

- Clearly, if Alice knows x, then Bob is convinced of her identity
- If Alice does not know x, she can guess c
- After t rounds, prob. of success = 2^{-t}
- If Alice does not know x but she succeeds, then we can factor

Alice guesses c=0

N, $y=x^2$ mod N

Pick r random
$w=r^2$ mod N

N, y

w →

Pick c = 0

← 0

z =r mod N

z →

z^2 = w mod N ?

TIME ↓

Alice guesses c=1

N, $y=x^2$ mod N

Pick z random
$w=z^2y^{-1}$ mod N

N, y

w →

Pick c = 1

← 1

z →

$z^2 = wy^c$ mod N ?
$wy^c = z^2y^{-1}y$ mod N

TIME ↓

Security

- If Alice does not know x but she succeeds
  - z_0, z_1
  - $z_0^2 = w$
  - $z_1^2 = wy$
  - $z_1/z_0 = \text{sqrt}(y)$
- But computing square roots is assumed to be as hard as factoring
  - If Alice compute square roots we can factor

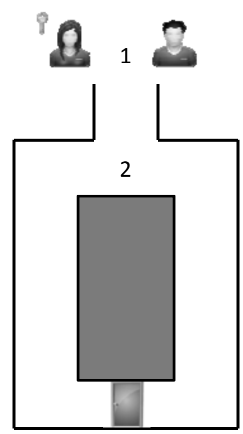
Zero-Knowledge

- FS Id scheme is ZK, i.e., it does not reveal information about x
- Proof
  - C = 0
    - Alice sends $w = r^2$ and $z = r$
    - Clearly no relation to x
  - C = 1
    - Alice sends $w = r^2$ and $z = rx$
    - rx is random
      - r is random
      - $\gcd(x, N) = 1$ (x can be any value in Z^*_N)
  - Assume that given **(N, $y = x^2$, $w = r^2$, rx)**, Bob computes x
    - He could do the same with (N, $y = x^2 \bmod N$)
    - He can choose a random $t = r^1 s \bmod N$ and compute

$$w^1 = t^2 y^{-1} = r_1^2 x^2 y^{-1} = r_1^2$$

Zero-Knowledge for Dummies (the cave)

1. Bob checks that door is locked and comes out to point 1 and looks away
2. Alice goes into the cave past point 2 (either right or left)
3. Bob looks into the cave from point 1
   1. Randomly picks right or left
   2. Shouts to Alice to come out from the picked direction
4. Alice moves to point 2
5. If Alice does not come out from the picked direction, Bob concludes Alice is a liar



Repeat n times