COM3020J - Protocols

Dr. Anca Jurcut
E-mail: anca.jurcut@ucd.ie

School of Computer Science and Informatics
University College Dublin,
Ireland

Timestamps

- A timestamp T is derived from current time
- Timestamps can be used to prevent replay
  - Used in Kerberos, for example
- Timestamps reduce number of msgs (good)
  - A challenge that both sides know in advance
- "Time" is a security-critical parameter (bad)
  - Clocks not same and/or network delays, so must allow for **clock skew** — creates risk of replay
  - How much clock skew is enough?

Public Key Authentication with Timestamp T

Alice

"I'm Alice", $\{\{T, K_{AB}\}K_{APriv}\}K_{BPub}$ →

← $\{\{T + 1, K_{AB}\}K_{BPriv}\}K_{APub}$

Bob

- Secure mutual authentication?
- Session key secure?
- Seems to be OK

Public Key Authentication with Timestamp T

Alice

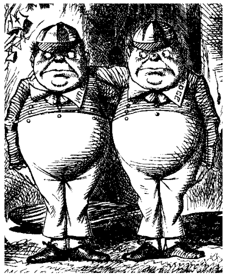
"I'm Alice", $\{\{T, K_{AB}\}K_{BPub}\}K_{APriv}$ →

← $\{\{T+1, K_{AB}\}K_{APub}\}K_{BPriv}$

Bob

- Secure authentication and session key?
- Trudy can use Alice's public key to find
  $\{T, K_{AB}\}K_{BPub}$ and then…

Public Key Authentication with Timestamp T



Trudy

“I’m Trudy”, $\{\{T, K_{AB}\}K_{BPub}\}K_{TrudyPriv}$ →

← $\{\{T+1, K_{AB}\}K_{TrudyPub}\}K_{BPriv}$

Bob

- Trudy obtains Alice-Bob session key K_{AB}
- **Note:** Trudy must act within clock skew

Public Key Authentication

- Sign and encrypt with nonce…
  - **Secure**
- Encrypt and sign with nonce…
  - **Secure**
- Sign and encrypt with timestamp…
  - **Secure**
- Encrypt and sign with timestamp…
  - **Insecure**
- Protocols can be subtle!

Public Key Authentication with Timestamp T

Alice

"I'm Alice", $\{\{T, K_{AB}\}K_{BPub}\}K_{APriv}$ →

← $\{\{T+1\}K_{APub}\}K_{BPriv}$

Bob

- Is this "encrypt and sign" secure?
  - Yes, seems to be OK
- Does "sign and encrypt" also work here?

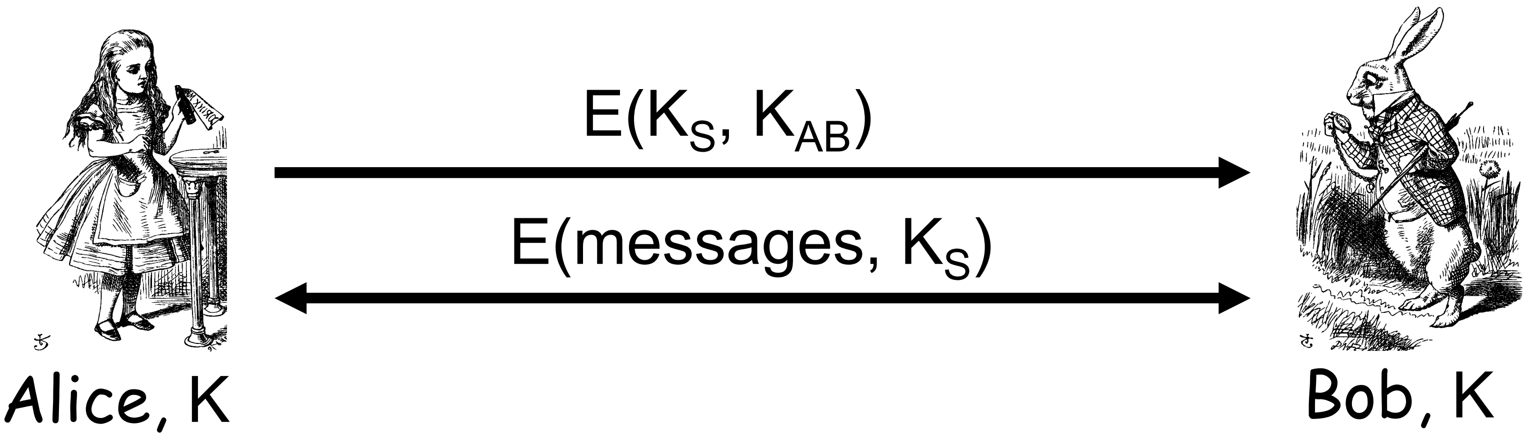
Perfect Forward Secrecy

- Consider this “issue”…
  - Alice encrypts message with shared key K_{AB} and sends ciphertext to Bob
  - Trudy records ciphertext and later attacks Alice’s (or Bob’s) computer to recover K_{AB}
  - Then Trudy decrypts recorded messages
- **Perfect forward secrecy (PFS):** Trudy cannot later decrypt recorded ciphertext
  - Even if Trudy gets key K_{AB} or other secret(s)
- Is PFS possible?

Perfect Forward Secrecy

- Suppose Alice and Bob share key K_{AB}
- For perfect forward secrecy, Alice and Bob cannot use K_{AB} to encrypt
- Instead they must use a session key K_S and forget it after it’s used
- Can Alice and Bob agree on session key K_S in a way that provides PFS?

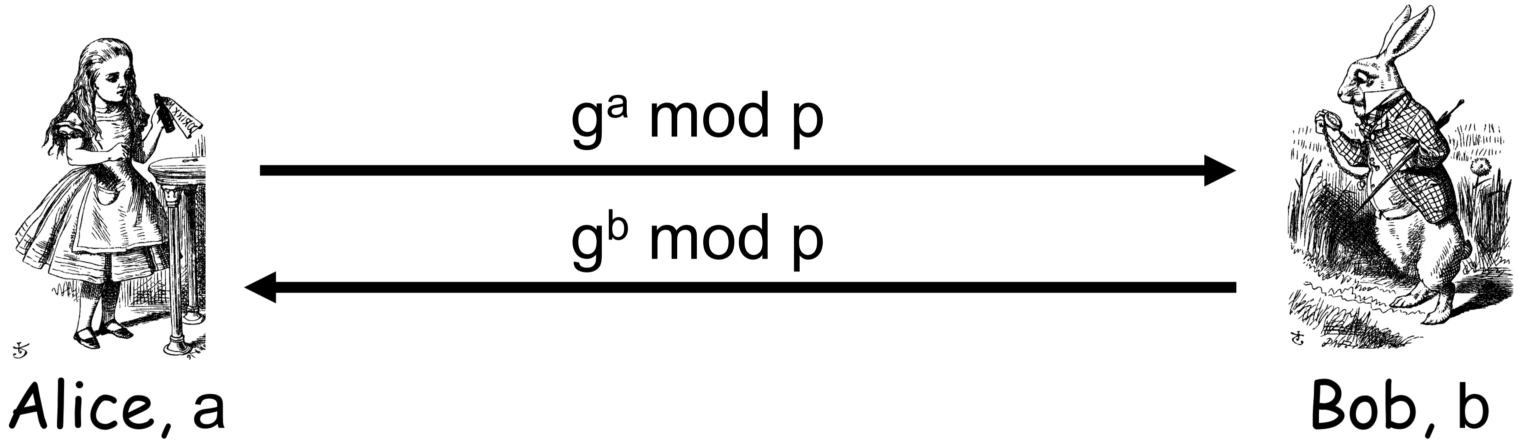
Naïve Session Key Protocol



- Trudy could record $E(K_S, K_{AB})$
- If Trudy later gets K_{AB} then she can get K_S
    - Then Trudy can decrypt recorded messages
- ***No*** perfect forward secrecy in this case

Perfect Forward Secrecy

- We can use **Diffie-Hellman** for PFS
- Recall: public g and p



- But Diffie-Hellman is subject to MiM
- How to get PFS and prevent MiM?

Perfect Forward Secrecy

$E(g^a \bmod p, K_{AB})$ →

← $E(g^b \bmod p, K_{AB})$

Alice: K_{AB}, a

Bob: K_{AB}, b

- Session key $K_S = g^{ab} \bmod p$
- Alice **forgets** a, Bob **forgets** b
- This is known as **Ephemeral Diffie-Hellman**
- Neither Alice nor Bob can later recover K_S
- Are there other ways to achieve PFS?

Mutual Authentication, Session Key and PFS

Alice

"I'm Alice", Na →

← Nb, $\{Na, g^b \bmod p\}K_{BPriv}$

$\{Nb, g^a \bmod p\}K_{APriv}$ →

Bob

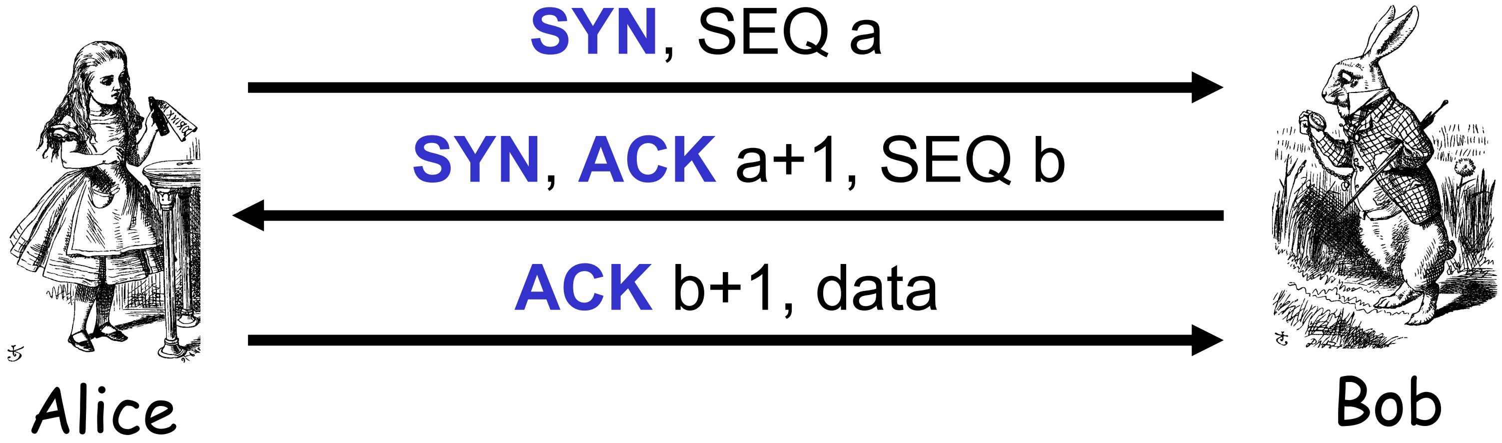
- Session key is $K_S = g^{ab} \bmod p$
- Alice forgets a and Bob forgets b
- If Trudy later gets Bob's and Alice's secrets, she cannot recover session key K_S
- **Note:** encryption is not required in this protocol. Signing the DH values prevents the MiM attack, while signing the nonces prevents a replay.

Authentication and TCP

TCP-based Authentication

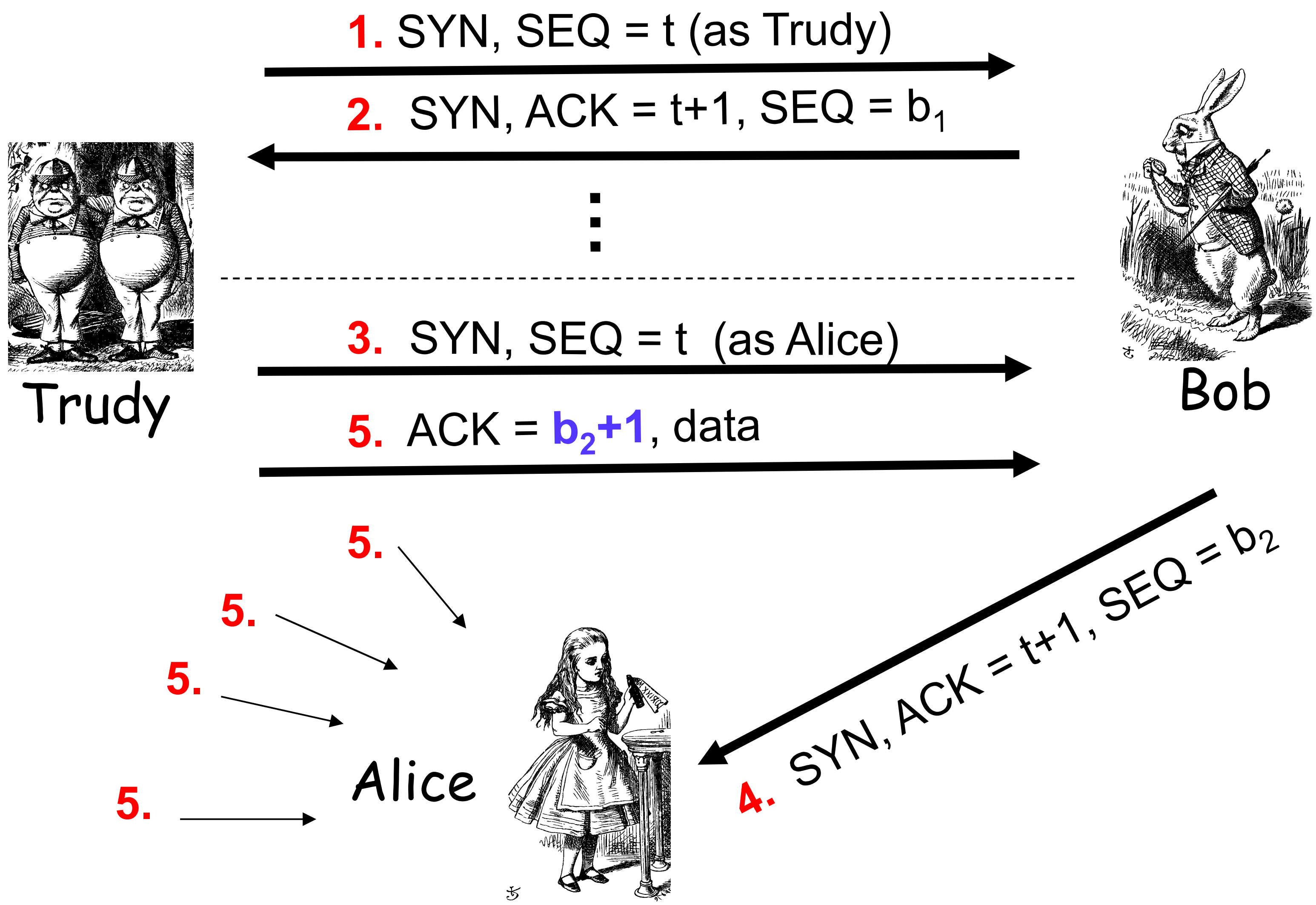
- TCP not intended for use as an authentication protocol
- But IP address in TCP connection may be (mis)used for authentication
- Also, one mode of IPSec relies on IP address for authentication

TCP 3-way Handshake

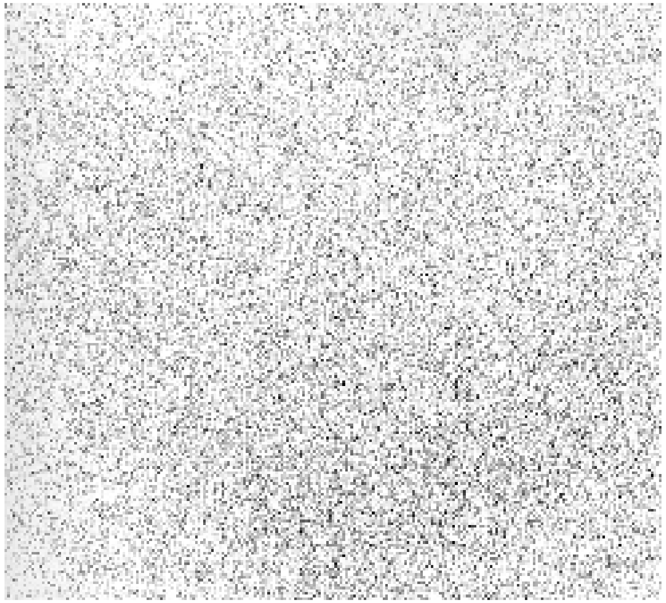


- Initial sequence numbers: SEQ a and SEQ b
  - Supposed to be selected at random
- If not, might have problems…

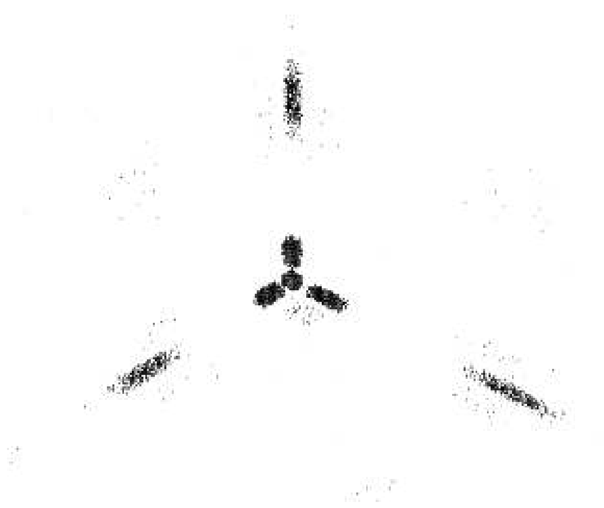
TCP Authentication Attack



TCP Authentication Attack



Random SEQ numbers



Initial SEQ numbers
Mac OS X

- If initial SEQ numbers not very random…
- …possible to guess initial SEQ number…
- …and previous attack will succeed

TCP Authentication Attack

- Trudy cannot see what Bob sends, but she can send packets to Bob, while posing as **Alice**
- Trudy must prevent Alice from receiving Bob's response (or else connection will terminate)
- If **password** (or other authentication) required, this attack fails
- If TCP connection is relied on for authentication, then attack might succeed
- **Bad idea** to rely on TCP for authentication

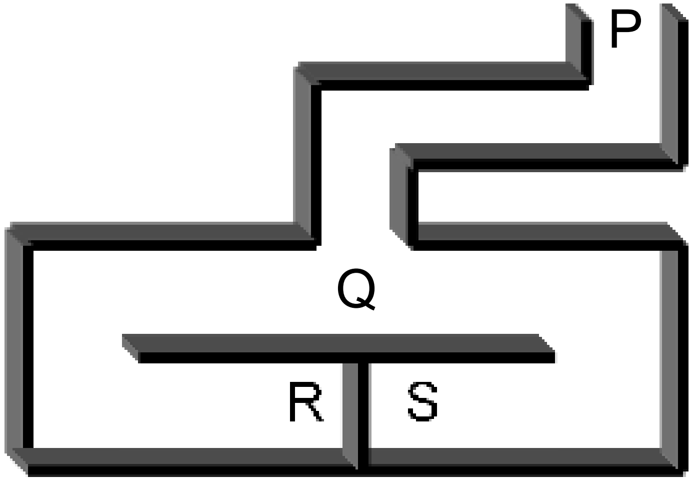
Zero Knowledge Proofs

Zero Knowledge Proof (ZKP)

- Alice wants to prove that she knows a secret without revealing **any** info about it
- Bob must verify that Alice knows secret
  - But, Bob gains no information about the secret
- Process is probabilistic
  - Bob can verify that Alice knows the secret to an arbitrarily high probability
- An “interactive proof system”

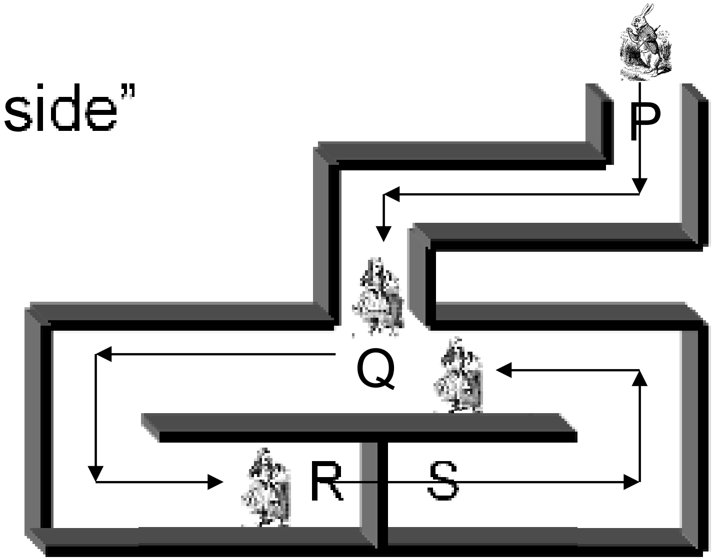
Bob's Cave

- Alice knows secret phrase to open path between R and S ("open sarsaparilla")
- Can she convince Bob that she knows the secret without revealing phrase?



Bob's Cave

- Bob: "Alice, come out on S side"
- Alice (quietly):
  "Open sarsaparilla"
- If Alice does not
  know the secret…



- …then Alice could come out from the correct side with probability 1/2
- If Bob repeats this n times and Alice does not know secret, she can only fool Bob with probability $1/2^n$

Fiat-Shamir Protocol

- Cave-based protocols are inconvenient
  - Can we achieve same effect without the cave?
- Finding square roots modulo N is difficult
  - Equivalent to factoring
- Suppose N = pq, where p and q prime
- Alice has a secret S
- N and $v = S^2 \bmod N$ are **public**, S is **secret**
- Alice must convince Bob that she knows S without revealing any information about S

Fiat-Shamir

Alice
secret S
random r

$x = r^2 \bmod N$ →

$e \in \{0,1\}$ ←

$y = r \cdot S^e \bmod N$ →

Bob
random e

- **Public:** Modulus N and $v = S^2 \bmod N$
- Alice selects random r, Bob chooses $e \in \{0,1\}$
- Bob verifies: $y^2 = x \cdot v^e \bmod N$
  - Note that $y^2 = r^2 \cdot S^{2e} = r^2 \cdot (S^2)^e = x \cdot v^e \bmod N$

Fiat-Shamir: e = 1

Alice
secret S
random r

$x = r^2 \bmod N$ →

← $e = 1$

$y = r \cdot S \bmod N$ →

Bob
random e

- **Public:** Modulus N and $v = S^2 \bmod N$
- Alice selects random r, Bob chooses e =1
- If $y^2 = x \cdot v \bmod N$ then Bob accepts it
  - And Alice passes this iteration of the protocol
- Note that Alice must know S in this case

Fiat-Shamir: e = 0

Alice
secret S
random r

$x = r^2 \bmod N$ →

← $e = 0$

$y = r \bmod N$ →

Bob
random e

- **Public:** Modulus N and $v = S^2 \bmod N$
- Alice selects random r, Bob chooses $e = 0$
- Bob must checks whether $y^2 = x \bmod N$
- "Alice" does **not** need to know S in this case!

Fiat-Shamir

- **Public:** modulus N and $v = S^2 \bmod N$
- **Secret:** Alice knows S
- Alice selects random r and **commits** to r by sending $x = r^2 \bmod N$ to Bob
- Bob sends **challenge** $e \in \{0,1\}$ to Alice
- Alice **responds** with $y = r \cdot S^e \bmod N$
- Bob checks whether $y^2 = x \cdot v^e \bmod N$
  - Does this prove response is from Alice?

Does Fiat-Shamir Work?

- If everyone follows protocol, math works:
  - Public: $v = S^2 \bmod N$
  - Alice to Bob: $x = r^2 \bmod N$ and $y = r \cdot S^e \bmod N$
  - Bob verifies: $y^2 = x \cdot v^e \bmod N$
- **Can Trudy convince Bob she is Alice?**
  - If Trudy expects $e = 0$, she follows the protocol: send $\mathbf{x = r^2}$ in msg 1 *and* $\mathbf{y = r}$ in msg 3
  - If Trudy expects $e = 1$, she sends $\mathbf{x = r^2 \cdot v^{-1}}$ in msg 1 and $\mathbf{y = r}$ in msg 3
- If Bob chooses $e \in \{0,1\}$ at random, Trudy can only trick Bob with probability 1/2

Fiat-Shamir Facts

- Trudy can trick Bob with probability 1/2, but…
  - …after n iterations, the probability that Trudy can convince Bob that she is Alice is only $1/2^n$
  - Just like Bob's cave!
- Bob's $e \in \{0,1\}$ must be unpredictable
- Alice must use new r each iteration, or else…
  - If e = 0, Alice sends **r mod N** in message 3
  - If e = 1, Alice sends **r $\cdot$ S mod N** in message 3
  - Anyone can find S given r mod N and r $\cdot$ S mod N

Fiat-Shamir Zero Knowledge?

- Zero knowledge means that nobody learns ***anything*** about the secret S
  - **Public:** $v = S^2 \bmod N$
  - Trudy sees $r^2 \bmod N$ in message 1
  - Trudy sees $r \cdot S \bmod N$ in message 3 (if $e = 1$)
- If Trudy can find r from $\mathbf{r^2 \bmod N}$, she gets S
  - But that requires modular square root calculation
  - If Trudy could find modular square roots, she could get S from **public** v
- Protocol does not seem to "help" to find S

ZKP in the Real World

- Public key certificates identify users
  - No anonymity if certificates sent in plaintext
- ZKP offers a way to authenticate without revealing identities
- ZKP supported in MS's Next Generation Secure Computing Base (NGSCB), where…
  - …ZKP used to authenticate software "without revealing machine identifying data"
- ZKP is **not** just pointless mathematics!

Best Authentication Protocol?

- It depends on…
  - The sensitivity of the application/data
  - The delay that is tolerable
  - The cost (computation) that is tolerable
  - What crypto is supported (public key, symmetric key, …)
  - Whether mutual authentication is required
  - Whether PFS, anonymity, etc., are concern
- …and possibly other factors