Announcement

Midterm: 5/9 (Tuesday)

- Closed book/notes, paper-based
- Only pens, pencils & erasers are allowed.
  - Pens are preferred for clarity
- No other items can be present = no notes, books, smartphones, tablets, laptops, earbuds, etc.
- Bring your ID with you for identity verification
- Sit far apart from each other

1

1

Lecture 8

Public Key Cryptography II: Signatures + Identification

[lecture slides are adapted from previous slides by Prof. Gene Tsudik]

2

2

Recall RSA encryption

$$Let\ n = pq \quad where \quad p,q - large\ primes$$
$$e,d \in Z^*_{\Phi(n)} \quad and \quad ed \equiv 1 \bmod \Phi(n)$$
$$where: \quad \Phi(n) = (p-1)(q-1) = pq - p - q + 1$$

$$Secrets: p,q,d$$

$$Publics: n,e$$

$$Encryption: message = m < n$$
$$E(m) = y = m^e \bmod n$$
$$Decryption: ciphertext = y$$
$$D(y) = m' = y^d \bmod n$$

3

More on RSA

- **Modulus n is unique per user →**
  - **2 or more parties cannot share the same n**
- **What happens if Alice and Bob share the same modulus?**
  - **Alice has (e',d',n) and Bob – (e",d",n)**
  - **Alice wants to compute d" (Bob's private key), but does not know phi(n)**
  - **She knows that: e' * d'= 1 mod phi(n)**
  - **So: e' * d' = k * phi(n) + 1 and: e' * d' - 1 = k * phi(n)**
  - **Alice just needs to compute inverse of e" mod X**
    - **where X = e' * d' – 1 = k * phi(n)**
    - **let's call this inverse d"'**
    - **and remember that: d"' * e" = k' * k * phi(n) + 1**
    - **can we be sure that: d"' = d" ?**
  - **Is it possible that e" has no inverse mod X?**
    - **Yes, if gcd(e",k)>1 but this is very, very UNLIKELY!**
  - **For all decryption purposes, d"' is EQUIVALENT to d"**
  - **Suppose Eve encrypted for Bob: C = $(m)^{e"}$ mod n**
  - **Alice computes:**

$$C^{d"'} \bmod n = m^{e"d"'} \bmod n = (m)^{k' * k * phi(n) + 1} \bmod n = m$$

4

4

El Gamal PK Cryptosystem (`83)

p – large prime
b – base, primitive element, generator
x – private exponent
y – public residue; $y \equiv b^x \bmod p$
$P = Z_p^*$
$C = Z_p^* \times Z_p^*$
publics : p, b, y
secrets : x

Encryption :
1. generate random $r \in Z_{p-1}$
2. compute : $k = b^r \bmod p$
3. compute : $c = my^r \bmod p = mb^{xr} \bmod p$
4. ciphertext = $\{k, c\}$

Decryption :
1. compute $k^x \bmod p$
2. compute $(k^x)^{-1} \bmod p$
3. $m' = (k^x)^{-1} c = b^{-rx} mb^{xr} \bmod p = m$

5

5

El Gamal (Example)

$p = 13$
$b = 2$
$x = 9$
$y = 2^9 \bmod 13 = 5$

Encryption:
$m = 11$
$r = 10$
$k = 2^{10} \bmod 13 = 10$
$c = 11 * 5^{10} \bmod 13 = 2$
ciphertext = $\{10, 2\}$

Decryption:
$10^9 \bmod 13 = 12$
$12^{-1} \bmod 13 = 12$
$2 * 12 = 24 \equiv 11 \bmod 13$

6

6

Summary: Public Key Cryptography

- Diffie-Hellman (DH) key pre-distribution protocol
  - **Not for encryption by itself** → it's for establishing shared secret K_{ab}
    - With a shared secret key, can then use symmetric crypto like AES
  - Security measured by $|p|$, i.e., # of bits in p, usually 2048/3072/…
  - Interactive key exchange is vulnerable to **MitM attack**
- RSA encryption
  - Public exponent e is usually **short** (common ones: 3, 5, 17, 257 or 65537)
  - Private exponent d is usually **very long** → may it hard to easily decrypt a message by brute-forcing the key
  - Security measured by $|n|$, where $n = pq$, e.g., RSA-2048 means $|n| = 2048$
  - Performance improvement: **square-and-multiply** to reduce exponentiation cost, **CRT** to accelerate encryption/decryption
  - **Lack of randomness** → vulnerable to replay attack
- El Gamal encryption
  - Based on DH, but for encryption
  - Added randomness to improve the security against replay attacks

Remember AES key is 128/192/256 bits

7

7

Digital Signatures

- Integrity
- Authentication
- Non-Repudiation
- Time-Stamping
- Causality
- Authorization



8

8

Digital Signatures

A signature scheme:

Usually, a message hash

(P,A,K,Sign,Verify)

***P* - plaintext (msgs)**

***A* - signatures**

***K* - keys**

Sign* - signing function: (*P*K)->A

Verify* - verification function: *(P*A*K)* → *{0,1}

9

9

Digital Signatures



- Integrity
- Authentication
- Non-Repudiation
- Time-Stamping
- Causality
- Authorization

10

10

Public Key Cryptography

- Asymmetric cryptography
- Invented in 1974-1978 (Diffie-Hellman and Rivest-Shamir-Adleman)
- Two keys: private (SK), public (PK)
  - Encryption: with public key;
  - Decryption: with private key
  - Digital Signatures: Signing by private key; Verification by public key. i.e., "encrypt" message digest/hash -- $h(m)$ -- with private key
    - Authorship (authentication)
    - Integrity: Similar to MAC
    - Non-repudiation: **can't do with symmetric key cryptography**
- Much **slower** than conventional cryptography
  - Often used together with conventional cryptography, e.g., to encrypt session keys

11

RSA Signature Scheme

Use the fact that, in RSA, encryption reverses "decryption"

Let n = pq where $p \neq q$ are two (large) primes

$e \in Z^*_{\Phi(n)}$ and $e = d^{-1} \bmod \Phi(n)$ and $ed \equiv 1 \bmod \Phi(n)$

$\Phi(n) = (p-1)(q-1)$

$Secrets : p, q, d$

$Publics : n, e$

Signing : $message = m < n$

$Sign(m) : y = m^d \bmod n$

Verification : $signature = y$

$Verify(y, m) : (m = y^e)$???

Recall RSA encryption

$Let\ n = pq \quad where \quad p,q - large\ primes$

$e,d \in Z^*_{\Phi(n)} \quad and \quad ed \equiv 1\ mod\ \Phi(n)$

$where: \quad \Phi(n) = (p-1)(q-1) = pq - p - q + 1$

$Secrets: p,q,d$

$Publics: n,e$

$Encryption: message = m < n$

$E(m) = y = m^e \bmod n$

$Decryption: ciphertext = y$

$D(y) = m' = y^d \bmod n$

13

Public Key Cryptography

- Asymmetric cryptography
- Invented in 1974-1978 (Diffie-Hellman and Rivest-Shamir-Adleman)
- Two keys: private (SK), public (PK)
  - Encryption: with public key;
  - Decryption: with private key
  - Digital Signatures: Signing by private key; Verification by public key. i.e., "encrypt" message digest/hash -- $h(m)$ -- with private key
    - Authorship (authentication)
    - Integrity: Similar to MAC
    - Non-repudiation: **can't do with symmetric key cryptography**
- Much **slower** than conventional cryptography
  - Often used together with conventional cryptography, e.g., to encrypt session keys

14

14

RSA Signature Scheme (contd)

- The Good:
  - Verification can be cheap (like RSA encryption)
  - Mechanically same as RSA decryption function
  - Security based on RSA encryption
  - Signing is harder but #verify-s > 1 …
- The Bad:
  - RSA is malleable: signatures can be "massaged"
    - $m_1^d * m_2^d = (m_1 * m_2)^d$
  - Phony "random" signatures
    - compute Y=RSA(e,X)=X^e mod n
    - X is a signature of Y because Y^d=X mod n

Plaintext	SIG
X^e ←	X

- The Ugly:
  - Signing requires integrity!

15

15

RSA Signature Scheme

Use the fact that, in RSA, encryption reverses "decryption"

Let n = pq where p ≠ q are two (large) primes

$e \in Z^*_{\Phi(n)}$ and $e = d^{-1} \bmod \Phi(n)$ and $ed \equiv 1 \bmod \Phi(n)$

$\Phi(n) = (p-1)(q-1)$

$Secrets : p, q, d$

$Publics : n, e$

Signing : $message = m < n$

$Sign(m) : y = m^d \bmod n$

Verification : $signature = y$

$Verify(y, m) : (m = y^e)$???

16

RSA Signature Scheme (contd)

- The Good:
  - Verification can be cheap (like RSA encryption)
  - Mechanically same as RSA decryption function
  - Security based on RSA encryption
  - Signing is harder but #verify-s > 1 …
- The Bad:
  - RSA is malleable: signatures can be "massaged"
    - $m_1^d * m_2^d = (m_1 * m_2)^d$
  - Phony "random" signatures
    - compute Y=RSA(e,X)=X^e mod n
    - X is a signature of Y because Y^d=X mod n

Plaintext	SIG
X^e	← X

- The Ugly:
  - Signing requires integrity!
  - How to sign multiple blocks when m > n?
  - Deterministic – needs additional randomization!

17

17

El Gamal Signature Scheme

$p - large\ prime$
$b - base, generator$
$x - private\ exponent$
$y - public\ residue;\ y \equiv b^x \bmod p$
$P = Z_p^*$
$A = Z_p^* \times Z_p^*$
$publics : p, b, y$
$secrets : x$

Signing :
1. $generate\ random\ r \in Z_{p-1}$
2. $compute : k = b^r \bmod p$
3. $compute : c = (m - xk) r^{-1} \bmod p - 1$
4. $signature = \{k, c\}$

Verifying :
$y^k k^c \bmod p = b^m \bmod p$???

notice that :
$y^k k^c = b^{xb^r} (b^r)^{(m/r - xk/r)} = b^{xb^r + m - xb^r} = b^m$

18

18

El Gamal PK Encryption Scheme

p – large prime
b – base, primitive element, generator
x – private exponent
y – public residue; $y \equiv b^x \bmod p$
$P = Z_p^*$
$C = Z_p^* \times Z_p^*$
publics : p, b, y
secrets : x

Encryption :
1. *generate random* $r \in Z^*_{p-1}$
2. *compute* : $k = b^r \bmod p$
3. *compute* : $c = my^r \bmod p = mb^{xr} \bmod p$
4. *ciphertext* $= \{k, c\}$

Decryption :
1. *compute* $k^x \bmod p$
2. *compute* $(k^x)^{-1} \bmod p$
3. $m' = (k^x)^{-1} c = b^{-rx} m b^{xr} \bmod p = m$

El Gamal Signature Scheme

p – large prime
b – base, generator
x – private exponent
y – public residue; $y \equiv b^x \bmod p$
$P = Z_p^*$
$A = Z_p^* \times Z_p^*$
publics : p, b, y
secrets : x

Signing :
1. *generate random* $r \in Z^*_{p-1}$
2. *compute* : $k = b^r \bmod p$
3. *compute* : $c = (m - xk) r^{-1} \bmod p - 1$
4. *signature* $= \{k, c\}$

Verifying :
$y^k k^c \bmod p = b^m \bmod p$???

notice that :
$y^k k^c = b^{xb^r} (b^r)^{(m/r - xk/r)} = b^{xb^r + m - xb^r} = b^m$

19

El Gamal Signature Scheme (cont'd)

The good:
- Signing is cheap(er)
- No phony signatures since it is "reversible" like RSA
- Non-deterministic (randomized)

The bad:
- Need GOOD source of random numbers
- Randomizers cannot be revealed (trace)
- Randomizers cannot be reused

20

20

The Digital Signature Standard (DSS)

- Why DSS?
- RSA issues: patents, malleability, etc.
- A variant of El Gamal, but better performance
- Originally for |p|=512 bits, now up to 1024
- Optimized for signature size (320- vs. 1024-bit)
- Signing - 1 exp, 1 inv, verification - 2 exps, 1 inv
- No attacks thus far

21

El Gamal

$p - large\ prime$
$b - base, generator$
$x - private\ exponent$
$y - public\ residue; y \equiv b^x \bmod p$
$P = Z_p^*, A = Z_p^* \times Z_p^*$
$publics: p, b, y \quad secrets: x$

Signing:
1. $generate\ random \quad r \in Z^*_{p-1}$
2. $compute: k = b^r \bmod p$
3. $compute: c = (m - xk) r^{-1} \bmod p - 1$
4. $signature = \{k, c\}$

Verifying:
$y^k k^c \bmod p = b^m \bmod p \quad ???$

DSS

$p - 512 - bit\ prime$
$q - 160 - bit\ prime, (p-1)\%q = 0$
$b - base, b^q \equiv 1 \bmod p \quad (b = \delta^{(p-1)/q})$
$x - private\ exponent$
$y - public\ residue; y \equiv b^x \bmod p$
$P = Z_p^*, A = Z_q \times Z_q$
$publics: p, q, b, y \quad secrets: x$

Signing:
1. $generate\ random \quad r \in Z^*_{q-1}$
2. $compute: k = (b^r \bmod p) \bmod q$
3. $compute: c = (m + xk) r^{-1} \bmod q$
4. $signature = \{k, c\}$

Verifying:
$(b^{mc^{-1}} y^{kc^{-1}} \bmod p) \bmod q = b^r \bmod p \quad ???$

notice that:
$b^{mc^{-1}} y^{kc^{-1}} = b^{mr/(m+xb^r)} (b^x)^{(b^r r/(m+xb^r))}$
$= b^{(mr+xb^r r)/(m+xb^r)} = b^r$

22

Other interesting constructions around our topic…

Interactive (Public) Key Exchange: Diffie-Hellman

Choose random v



$y_a = a^v \bmod p$

$y_b = a^w \bmod p$



Choose random w, Compute $K_{ba} = (y_a)^w \bmod p$

Compute $K_{ab} = (y_b)^v \bmod p$

Secure communication with K_{ab}



Eve is passive …

Use symmetric crypto to exchange keys?

25

25

Merkle's Puzzles (1974)





for $0 < i < 2^n = N$

Pick random values $X_i, Y_i, |X_i| \geq 2^* |Y_i|$, *where* $|Y_i| = n$

Pick random $index_i, |index_i| = n$

Form Puzzle $P_i = E(Y_i, \{index_i, Xi, S\})$

where S is a fixed string, e.g., "Alice to Bob"

$\{P_i \mid 0 < i < 2^n\}$ →

Pick random $j, 0 < j < 2^n$

Select P_j

Break Y_j *by brute force*

Obtain $\{index_j, X_j, S\}$

← $index_j$

Lookup $index_j$

Obtain X_j

Encrypted communication with X_j

Alice's effort: O(2^n)

Bob's effort: O($2^{|Yj|}$) = O(2^n)



Eve's effort:
O($2^n * 2^{|Yi|}$) = O($(2^n)^2$) = O($2^{|Xi|}$)

Such quadratic gap is best possible if using symmetric crypto [IR'89, BM'09]

26



Merkle's Puzzles (1974)



for $0 < i < 2^n = N$

Pick random values $X_i, Y_i, |X_i| \geq 2^*|Y_i|$, *where* $|Y_i| = n$

Pick random $index_i, |index_i| = n$

Form Puzzle $P_i = E(Y_i, \{index_i, Xi, S\})$

where S is a fixed string, e.g., "Alice to Bob"

Takeaway: Smart construction, but DH is more practical

$\{P_i \mid 0 < i < 2^n\}$

Pick random $j, 0 < j < 2^n$

Select P_j

Break Y_j *by brute force*

Obtain $\{index_j, X_j, S\}$

$index_j$

Lookup $index_j$

Obtain X_j

Encrypted communication with X_j

Alice's effort: O(2^n)

Bob's effort: O(2^{|Yj|}) = O(2^n)



Eve's effort:
O(2^n * 2^{|Yi|}) = O((2^n)^2) = O(2^{|Xi|})

Such quadratic gap is best possible if using symmetric crypto [IR'89, BM'09]

27

Other use of public key crypto (except encryption & signature)?

28

28

Identification/Authentication

- Identification/authentication is an interactive protocol whereby one party: "prover" (who claims to be, say, Alice) convinces the other party: "verifier" (Bob) that she is indeed Alice
- Identification/authentication can be accomplished with public key digital signatures
  - However, signatures carry information about private key
    - E.g., anyone who has Alice's signature can use it to prove that he/she is Alice
- Can we provide identification/authentication without revealing any info about the secret?
  - ***Zero-knowledge proof***: prove ownership of a secret without revealing any info about the secret

29

29

The Cave Analogy of Zero-Knowledge



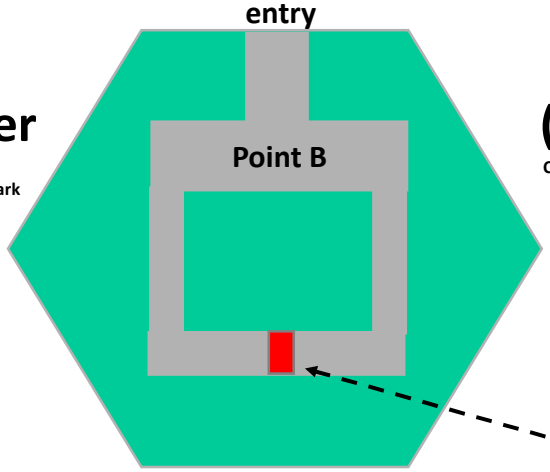
(V)erifier

Claustrophobic and afraid of the dark



(P)rover

Claims to have the key but won't show it



V cannot follow P into the cave

Locked door on both sides

30

30

The Cave Analogy of Zero-Knowledge

The Protocol:

1) V asks someone he trusts to check that the door is locked on both sides.

2) P goes into the maze past point B (heading either right or left)

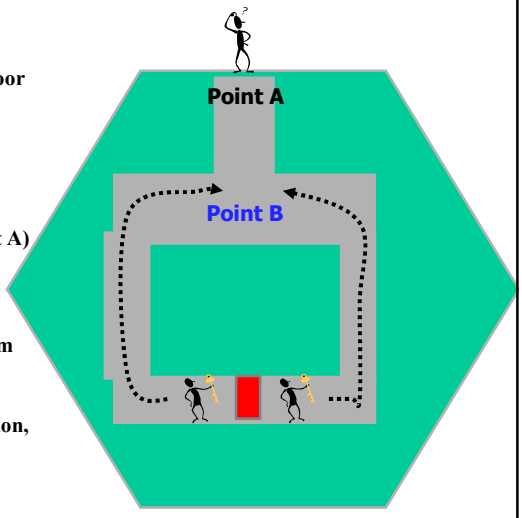
3) V looks into the cave (while standing at point A)

4) V randomly picks right or left

5) V shouts (very loudly!) for P to come out from the picked direction

6) If P doesn't come out from the picked direction, V knows that P is a liar and protocol terminates

REPEAT steps (2)-(6)
k TIMES



Someone do not have the key cannot always do both!

31

31

Fiat-Shamir Identification Scheme

- In Fiat-Shamir, prover has an RSA-like modulus **n = pq** where p and q are large primes and factorization of n is secret
- Secret Key: Prover (P) chooses a random value

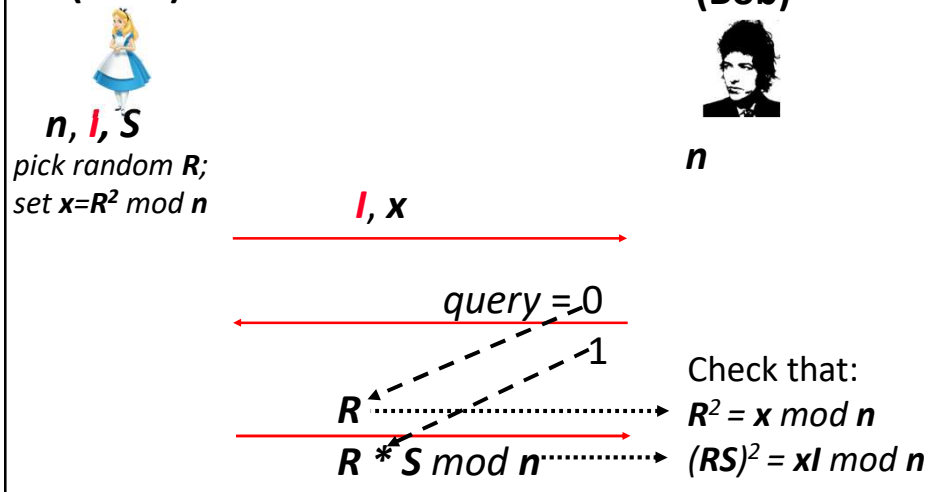
  $1 < S < n$ (to serve as the key)

  such that $\gcd(S,n) = 1$
- Public Key: P computes $I=S^2 \bmod n$, publishes (I,n) as his public key.
  - Assumption: Finding square roots mod n is at least as hard as factoring n
- Purpose of the protocol: P has to convince verifier (V) that he knows the secret S corresponding to the public key (I,n),
  - i.e., to prove that he knows a square root of I mod n, without revealing S or any portion thereof

32

32

Fiat-Shamir



33

Fiat-Shamir Identification Scheme

V wants to authenticate identity of P, who claims to have a public key I. Thus, V asks P to convince him that P knows the secret key S corresponding to I .

1. P chooses at random $1 < R < n$ and computes: $X = R^2 \bmod n$
2. P sends X to V
3. V randomly requests from P one of two things (0 or 1):
   (a) R
   or
   (b) RS mod n
4. P sends requested information

34

Fiat-Shamir ZK Identification Scheme

5. V checks the correct answer:
   a) R^2 ?= X (mod n)
      or
   b) $(R*S)^2$?= X*I (mod n)
6. If verification fails, V concludes that P does not know S
7. Protocol is repeated t (usually 20, 30, or log n) times, and, if each one succeeds, V concludes that P is the claimed party.

35

35

Fiat-Shamir Identification Scheme

CLAIM: Protocol does not reveal ANY information about S,

or The Fiat-Shamir protocol is **ZERO-KNOWLEDGE**

Proof: We show that no information on S is revealed:

- Clearly, when P sends X or R, it does not reveal any information about S
- When P sends *RS mod n*:
  - **RS mod n** is random, since R is random and gcd(S, n) = 1.
  - If adversary can compute any information about S from

    I, n, X and RS mod n

    it can also compute the same information on S from **I and n**, since it can choose a random number T, compute $X' = T^2I^{-1} = ((T/S)^2S^2I^{-1} = (T/S)^2$, then have: I, n, X', and (T/S)S mod n -> get S

    -> contradictory to the fact that we cannot easily find S given only I and n

36

36

Security

Clearly, if P knows S, then V is convinced of P's identity

If P does not know S, it can either:

1. Choose to send $x=R^2 \bmod n$, but in this case cannot provide $RS \bmod n$ → CANNOT answer query = 1

   or

2.

If Prover wants to answer query = 0 correctly





n, ***I*** *(doesn't know S)*

pick random ***R****;*
set $x=R^2 \bmod n$

I, ***x*** →

n

query = 0 ←

R →

Check that:
$R^2 = x \bmod n$

If Prover wants to answer query = 0 correctly





n, ***I*** *(doesn't know S)*

n

pick random ***R****;*
set ***x***=***R***^{2} *mod* ***n***

I, ***x***

query = 1

Cannot *provide correct (****R*S*** *mod* ***n)!***

Check that:
(***RS***)^2 = ***xI*** *mod* ***n***

39

Security

Clearly, if P knows S, then V is convinced of P's identity

If P does not know S, it can either:

1. Choose to send ***x***=***R***^{2} *mod* ***n***, but in this case cannot provide ***RS mod n*** → CANNOT answer query = 1

   or

2. Choose to send ***x***=***R***^{2}****I*** *mod* ***n***, so that can answer query = 1 correctly.

40

If Prover wants to answer query = 1 correctly





n, ***I*** *(doesn't know S)*
pick random ***R****;*
set ***x****=****R****²*****I*** *mod* ***n***

n

I, ***x=R²*I***

query = 1

Check that:
$(R*I)^2 = x*I \bmod n$

R*I *mod* ***n***

(Instead of: R*S *mod* ***n)***

41

41

If Prover wants to answer query = 1 correctly





n, ***I*** *(doesn't know S)*
pick random ***R****;*
set ***x****=****R****²*****I*** *mod* ***n***

n

I, ***x=R²*I***

query = 0

Cannot *provide correct*
(R' = R*S *mod* ***n)!***

Check that:
$(R')^2 = x \bmod n$

42

42

Security

Clearly, if P knows S, then V is convinced of P's identity

If P does not know S, it can either:

1. Choose to send $x=R^2 \bmod n$, but in this case cannot provide $RS \bmod n$ → CANNOT answer query = 1

   or

2. Choose to send $x=R^2 * I \bmod n$, so that can answer query = 1 correctly. But, in this case, P cannot answer query = 0, since to do so, needs to know S

43

43

Security

- In any case, adversary cannot answer both questions
- Since P does not know in advance (when choosing R or RS mod n) which question that V will ask, he cannot foresee the required choice. He can succeed in guessing V's question with probability 1/2 for each question.
- The probability that V fails to catch P in all runs is thus: 2^{-t}
  - e.g., 1 in 1,000,000,000 for t=20

44

44

Fiat-Shamir Identification Scheme

- In Fiat-Shamir, prover has an RSA-like modulus **n = pq** where p and q are large primes and factorization of n is secret
- Primes themselves are not used in the protocol
  - Unlike RSA, a trusted center can generate a global **n**, used by everyone, as long as nobody knows its factorization. Trusted center can then "forget" the factorization after computing **n**