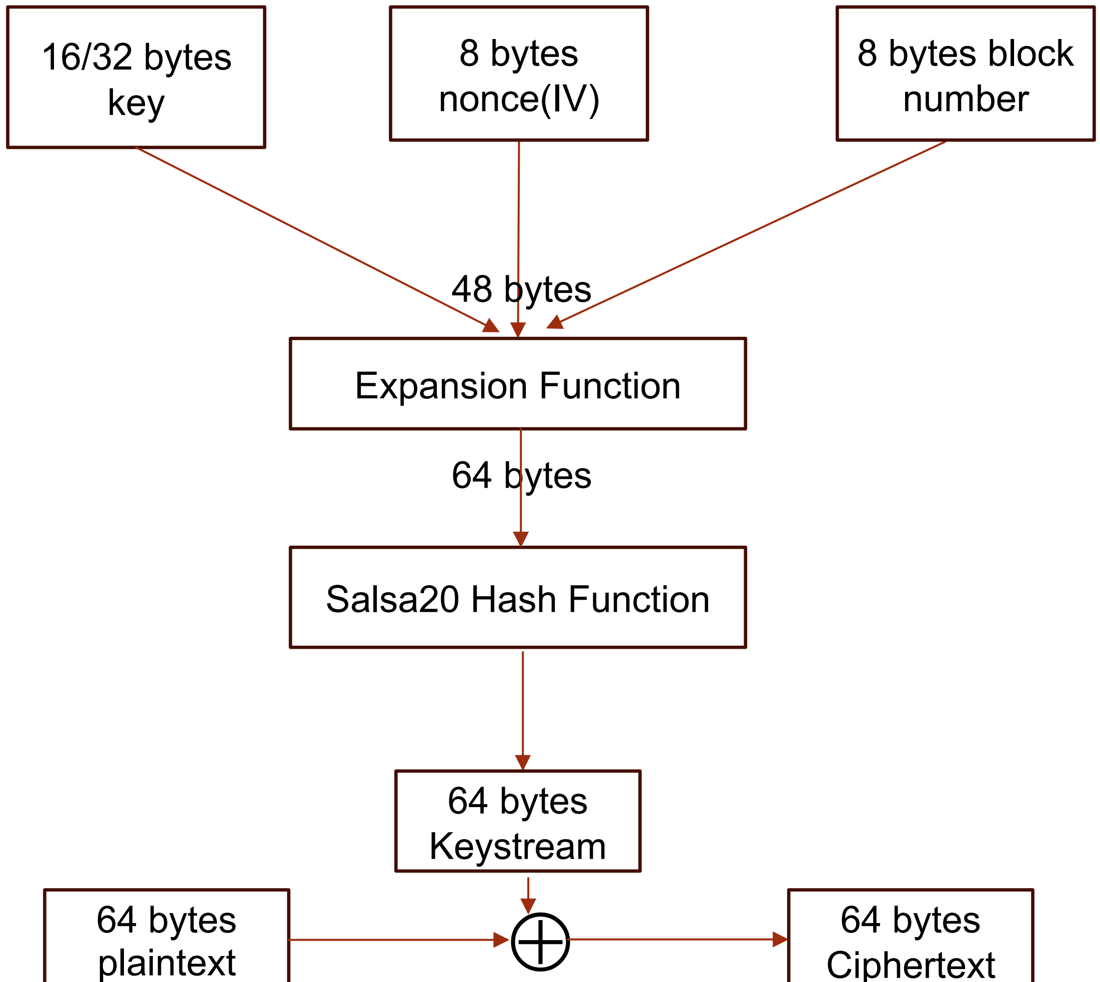
Cryptology

Salsa20 Encryption



BINGHAMTON UNIVERSITY
STATE UNIVERSITY OF NEW YORK

Salsa20 Encryption – Expansion

Initial state of Salsa20

4 X 4 matrix

"expa"	Key	Key	Key
Key	"nd 3"	Nonce	Nonce
Pos.	Pos.	"2-by"	Key
Key	Key	Key	"te k"

Each word is **4 bytes**

Eight words of key
Two words of nonce (IV)
Two words of stream position(block number)
Four fixed words

Constant word "expand 32-byte k" in ASCII

Salsa20

Initial state of Salsa20

"expa"	Key	Key	Key
Key	"nd 3"	Nonce	Nonce
Pos.	Pos.	"2-by"	Key
Key	Key	Key	"te k"

9 The Salsa20 expansion function

Inputs and outputs

If k is a 32-byte *or* 16-byte sequence and n is a 16-byte sequence then $\mathrm{Salsa20}_k(n)$ is a 64-byte sequence.

Let k be a 32-byte *or* 16-byte sequence. Let v be an 8-byte sequence. Let m be an ℓ-byte sequence for some $\ell \in \{0, 1, \ldots, 2^{70}\}$. The **Salsa20 encryption of** m **with nonce** v **under key** k, denoted $\mathrm{Salsa20}_k(v) \oplus m$, is an ℓ-byte sequence.

$\mathrm{Salsa20}_k(v)$ is the 2^{70}-byte sequence

Block number increased after each block

v is nonce

Block number is Pos

$$\mathrm{Salsa20}_k(v, \underline{0}), \mathrm{Salsa20}_k(v, \underline{1}), \mathrm{Salsa20}_k(v, \underline{2}), \ldots, \mathrm{Salsa20}_k(v, \underline{2^{64} - 1}).$$

Here $\underline{i}$ is the unique 8-byte sequence $(i_0, i_1, \ldots, i_7)$ such that $i = i_0 + 2^8 i_1 + 2^{16} i_2 + \cdots + 2^{56} i_7$.

Quiz

Quiz 1

Which of the following is true?

- **A: Digital signature can be used to achieve data integrity and non-repudiation**
- B: Symmetric keys can be used to achieve non-repudiation
- **C: Asymmetric key crypto can be used to achieve non-repudiation**
- D: When you receive a message signed by Alice encrypted with your public key, she must be Alice

Quiz 2

Which of the following is true?

- A: A public key certificate that CA 1 signs for Alice includes Alice's name, Alice's public key, and CA 1's public key
- B: A Certificate Authority (CA) can decrypt all messages encrypted with public keys it has signed, because it issues and signs the certificates
- C: When you receive from a person a public key certificate signed by CA 1 for Alice, you know that she must be Alice
- **D: The monopoly PKI trust model has a single point of failure**

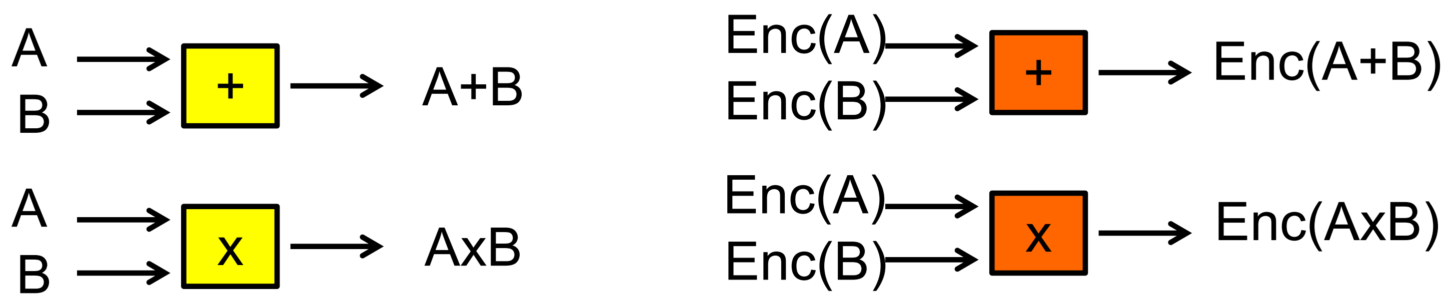
Quiz 3

What is true about cryptographic hash function?

- **A: A good cryptographic hash function h should make it difficult to find such x and y that h(x) = h(y)**
- B: Strong collision resistance means that given x and h(x), it is infeasible to find y ≠ x such that h(y) = h(x)
- **C: If a crypto hash function satisfies the strong collision resistance property, it also satisfies the weak collision resistance property**
- **D: MD5 and SHA-1 are the two most popular cryptographic hash functions**

Topics not covered…

- **Advanced cryptoanalysis**
  - Linear and differential cryptoanalysis (DES, 3DES)
  - Lattice reduction attack against the Knapsack cryptosystem
  - RSA timing attack
- **Homomorphic cryptography: useful in cloud computing**



  - **Partial** homomorphic cryptography (addition or multiplication)
  - **Fully** homomorphic crypography (both + and x)

General recommendations on crypto

- Use only standardized algorithms and protocols
  - No security through obscurity!
- Use existing, high-level crypto libraries (such as OpenSSL) and avoid writing your own low-level crypto
- Don’t use the same key for multiple purposes
  - E.g., encryption/MAC, or RSA encryption/signatures
- Use good random-number generation

Random Numbers in Cryptography

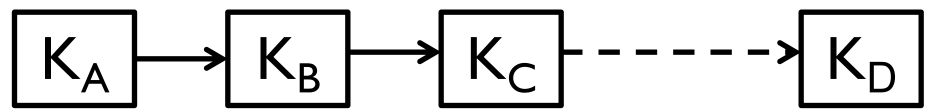
Random Numbers

- Random numbers used to generate **keys**
  - Symmetric keys
  - RSA: Prime numbers
  - Diffie Hellman: secret values

- Random numbers also used in simulations, statistics, etc.
  - Such numbers need to be "statistically" random

Random Numbers

- Cryptographic random numbers must be statistically random and **unpredictable**
- Suppose server generates symmetric keys sequentially
  - Alice: K_A
  - Bob: K_B
  - Charlie: K_C
  - Dave: K_D



- But, Alice, Bob, and Charlie don't like Dave
- Alice, Bob, and Charlie working together must not be able to determine K_D

Non-random Random Numbers

- Online version of Texas Hold 'em Poker
  - ASF Software, Inc.

- Random numbers used to shuffle the deck
- Program did not produce a random shuffle
- A serious problem or not?

Card Shuffle

- There are $52! > 2^{225}$ possible shuffles
- The poker program used “random” 32-bit integer to determine the shuffle
  - So, only 2^{32} distinct shuffles could occur
- Code used pseudo-random number generator (PRNG): $\text{randomize}()$
- Seed value for PRNG was function of number of milliseconds since midnight
- Less than 2^{27} milliseconds in a day (24 * 60 * 60 * 1000)
  - So, less than 2^{27} possible shuffles

Card Shuffle

- By synchronizing clock with server at the second level, number of shuffles that need to be tested $< 2^{18}$

- Could then test all 2^{18} in real time
  - Test each possible shuffle against “up” cards
  - Determine the actual shuffle for the hand

- Attacker knows **every card** after the first set of community cards were reveald

Poker Example

- Poker program is an extreme example
  - But common PRNGs are predictable
  - Only a question of how many outputs must be observed before determining the sequence
- Crypto random sequences not predictable
  - For example, keystream from RC4 cipher
  - But “seed” (or key) selection is still an issue!
- How to generate initial **random** values?
  - Keys (and, in some cases, seed values)

What is Random?

- True "randomness" hard to define
- **Entropy** is a measure of randomness
- Good sources of "true" randomness
  - Radioactive decay — radioactive computers are not too popular
  - Hardware devices — many good ones on the market
  - Lava lamp — relies on chaotic behavior

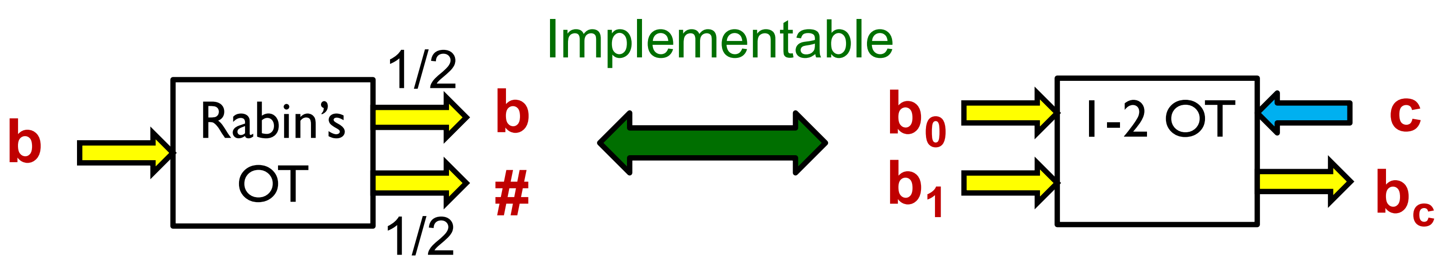
Randomness

- Sources of randomness via software
  - Software is (hopefully) deterministic
  - So must rely on external “random” events
  - Mouse movements, keyboard dynamics, network activity, etc.
- Can get **quality** random bits by such methods
- But **quantity** of bits is very limited
- Bottom line: “The use of pseudo-random processes to generate secret quantities can result in pseudo-security”

Oblivious Transfer

Oblivious Transfer

- Oblivious transfer is a type of protocol in which a sender transfers one of potentially many pieces of information to a receiver, but without knowing what piece has been transferred.
- Rabin's oblivious transfer: Alice chooses as input one bit b. Then, with probability 1/2, Bob gets the bit b, and nothing otherwise.
- 1-2 (1-out-of-2) oblivious transfer: Alice chooses as input two bits b_0 and b_1. Bob chooses a selection bit c and gets as output the bit b_c.



BINGHAMTON UNIVERSITY
STATE UNIVERSITY OF NEW YORK

Why use oblivious transfer?

- Privacy
  - Sender does not know receiver's choice, and receiver only learns one message
- Building block for multiparty computation(SMC)
  - Compute a function over their private inputs without revealing them to each other
- Efficiency
  - More efficient than other techniques that might achieve similar goals

1-2 (1-out-of-2) Oblivious Transfer

- Alice has two messages m_0 and m_1
- Bob has one bit b, and wishes to receive m_b
- Alice needs to ensure that Bob receives only one of the messages, but doesn't know which message Bob receives

1-2 OT (RSA)

- Alice has two messages to be sent, m_0 and m_1

m_0 and m_1

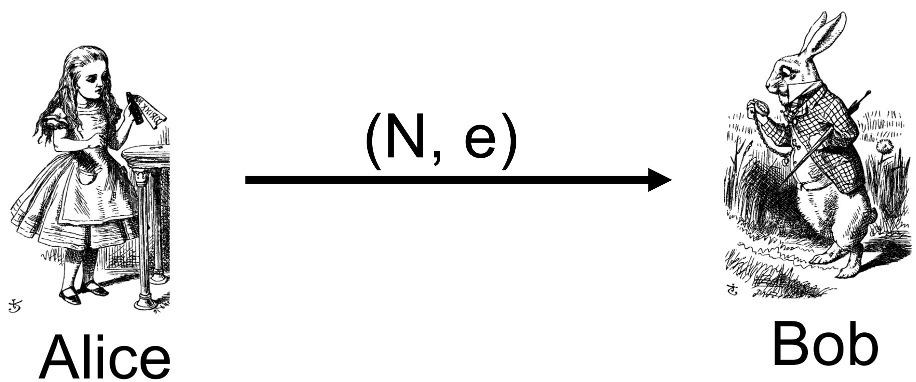
Alice

Bob

Binghamton University State University of New York

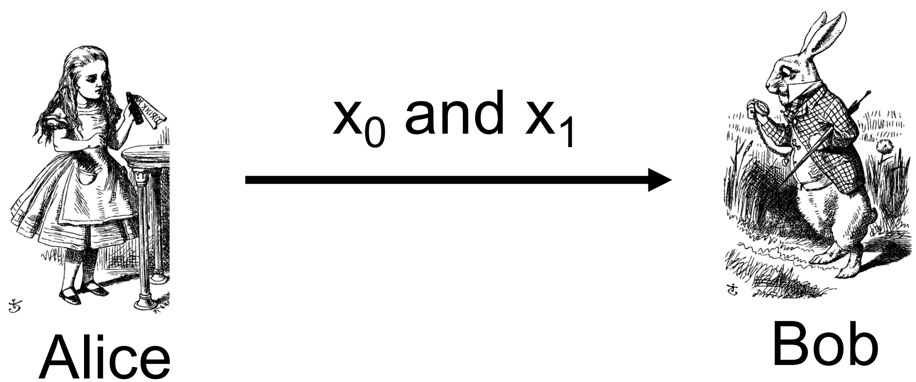
1-2 OT (RSA)

- Alice generates RSA key pair
  - Public key: (N, e)
  - Private: d
- Alice sends public key to Bob



1-2 OT (RSA)

- Alice generates two random messages, x_0 and x_1
- Alice sends both messages to Bob



1-2 OT (RSA)

- Bob chooses b in {0, 1}
- Bob generates random number k

b, k

Alice

Bob

BINGHAMTON UNIVERSITY
STATE UNIVERSITY OF NEW YORK

1-2 OT (RSA)

- Bob encrypts k with Alice's public key, use it blind x_b, and then sends it to Alice

$$v = (x_b + k^e) \bmod N$$

Alice

Bob

1-2 OT (RSA)

- Alice calculates: k_0 and k_1 (Alice doesn't know b)
  - $k_0 = (v - x_0)^d \bmod N$
  - $k_1 = (v - x_1)^d \bmod N$

$$v = (x_b + k^e) \bmod N$$

Alice

Bob

1-2 OT (RSA)

- Alice sends both messages to Bob
  - $m_0' = m_0 + k_0$
  - $m_1' = m_1 + k_1$

$$k_0 = (v - x_0)^d \bmod N$$
$$k_1 = (v - x_1)^d \bmod N$$

Alice

$$\xrightarrow{m_0',\ m_1'}$$

Bob

1-2 OT (RSA)

- Bob decrypts $m_b = m_b' - k$ since he knows which message he has chosen (i.e., b) and also k

$$\boxed{\begin{aligned} m_0' &= m_0 + k_0 \\ m_1' &= m_1 + k_1 \end{aligned}}$$

Alice

Bob

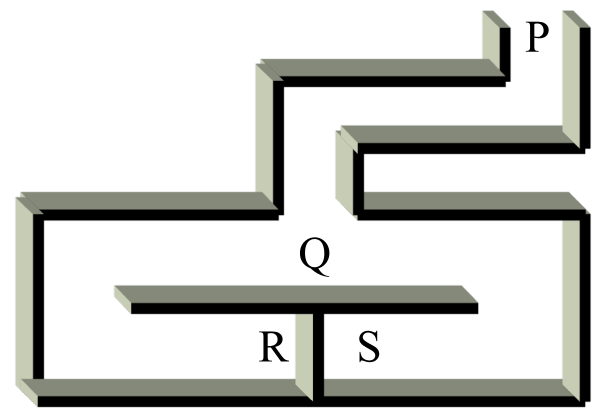
Zero Knowledge Proofs

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
  - Sounds impossible!
- Process is probabilistic
  - Bob can verify that Alice knows the secret to an arbitrarily high probability
- An "interactive proof system"

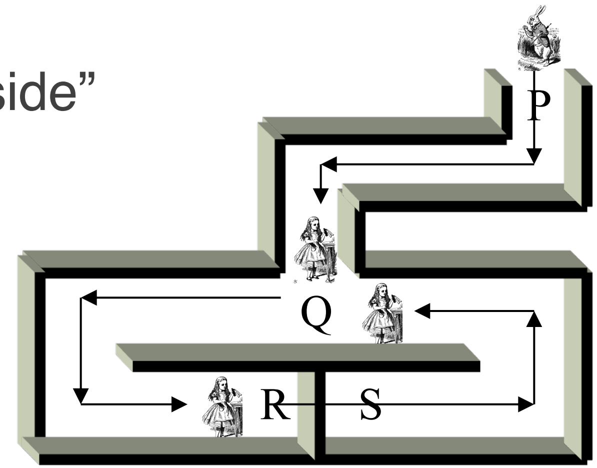
Bob's Cave

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



Bob's Cave

- Bob: "Alice, come out on S side"

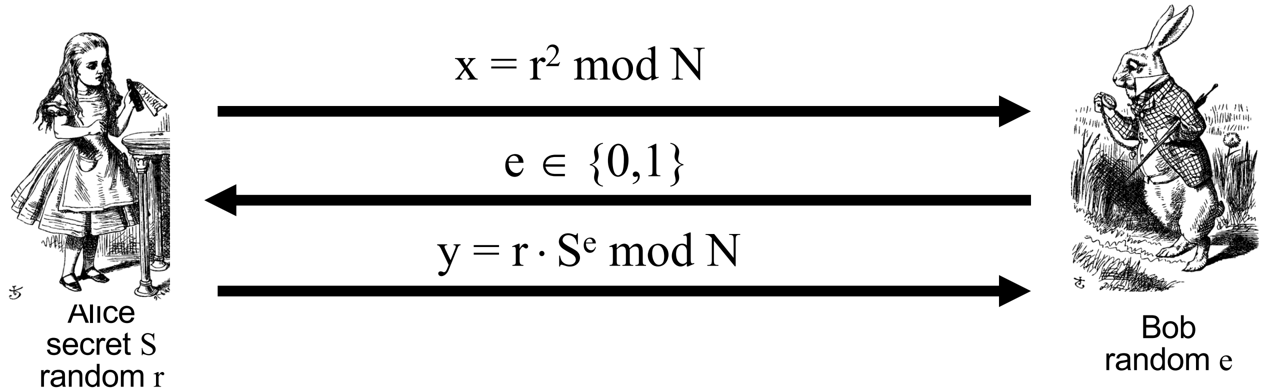


- Alice (quietly): "Open sesame"
- If Alice does not know the secret…
- …then Alice could come out from the correct side with probability $1/2$
- If Bob repeats this n times and Alice does not know secret, she can only fool Bob with probability $1/2^n$

Fiat-Shamir Protocol

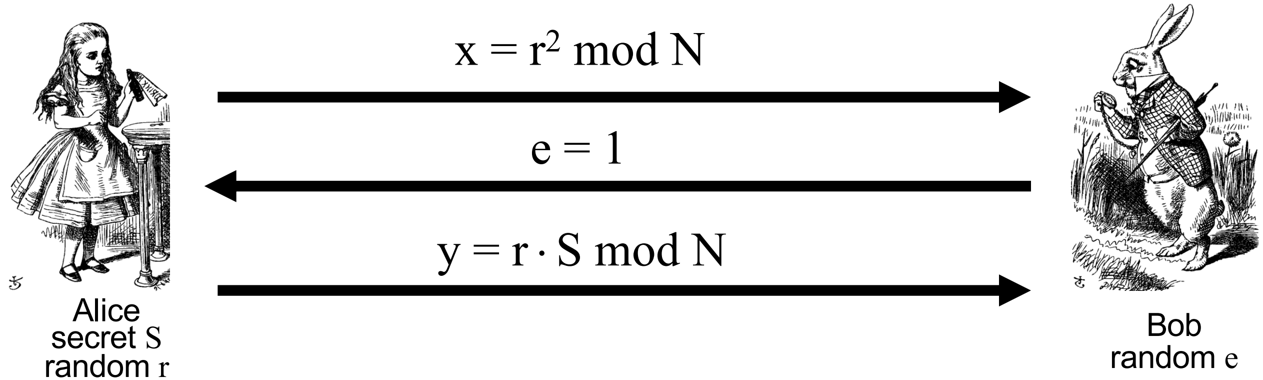
- Cave-based protocols are inconvenient
  - Can we achieve same effect without the cave?
- Finding square roots modulo N is difficult
  - Equivalent to factoring
- Suppose $N = pq$, where p and q prime
- Alice has a secret S
- N and $v = S^2 \bmod N$ are **public**, S is **secret**
- Alice must convince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

Fiat-Shamir



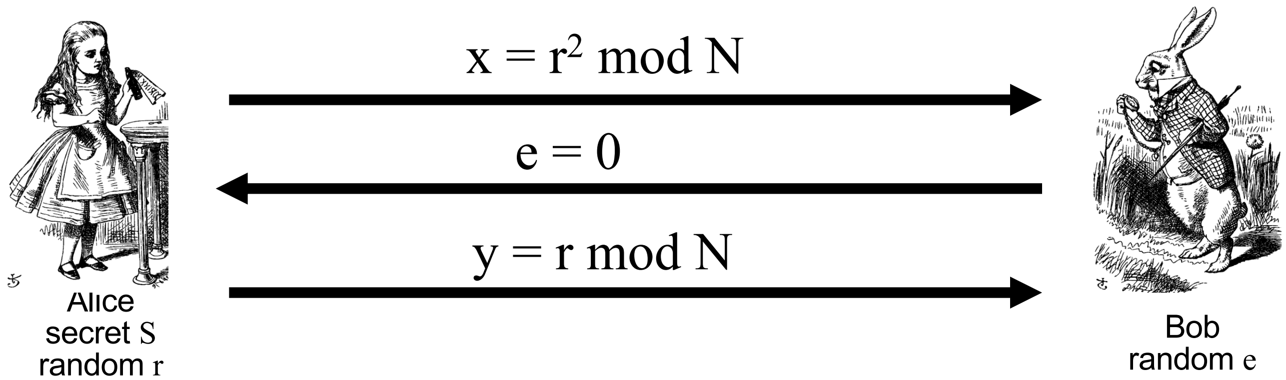
- **Public:** Modulus N and $v = S^2 \bmod N$
- Alice selects random r, Bob chooses $e \in \{0,1\}$
- Bob verifies: $\mathbf{y^2 = x \cdot v^e \bmod N}$
    - Note that $y^2 = r^2 \cdot S^{2e} = r^2 \cdot (S^2)^e = x \cdot v^e \bmod N$

Fiat-Shamir: e = 1



- **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



- **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!
  - Same as Alice happened to come out from right side in the cave

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 $x = r^2$ in msg 1 and $y = r$ in msg 3
  - If Trudy expects $e = 1$, she sends $x = r^2 \cdot v^{-1}$ in msg 1 and $y = r$ 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 $r \bmod N$ in message 3
  - If $e = 1$, Alice sends $r \cdot S \bmod N$ in message 3
  - Anyone can find S given $r \bmod N$ and $r \cdot S \bmod N$

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 $r^2 \bmod N$, 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

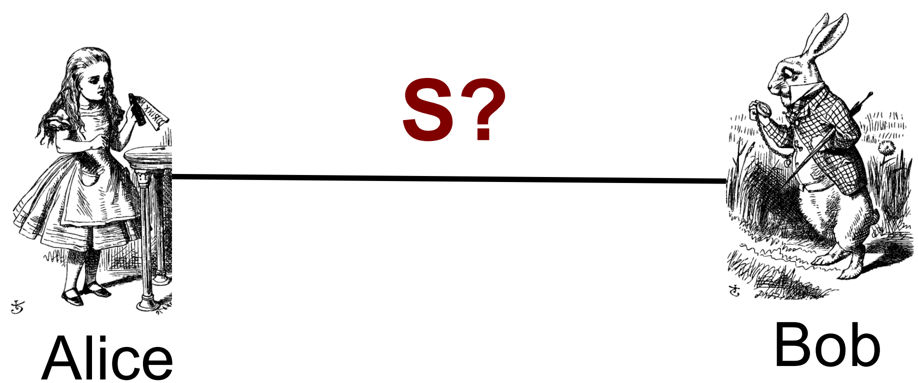
ZKP in the Real World

- 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!

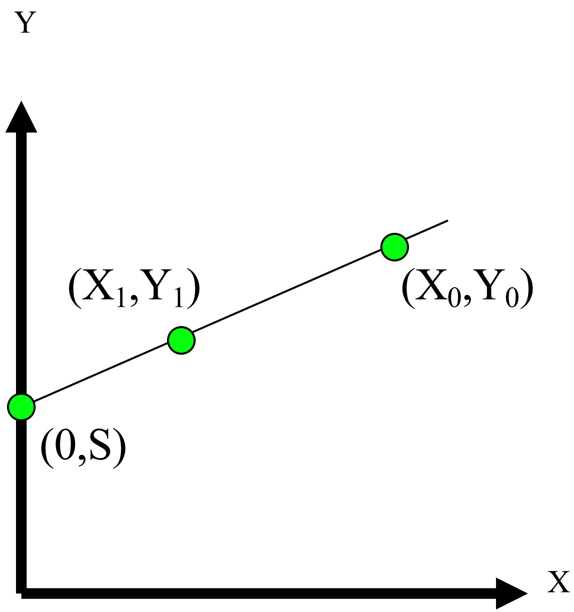
Secret Sharing

What is secret sharing

- Goal: Alice and Bob want to share a secret S in the sense that:
  - Neither Alice nor Bob alone (nor anyone else) can determine S with a probability better than guessing
  - Alice and Bob together can easily determine S



Shamir's Secret Sharing

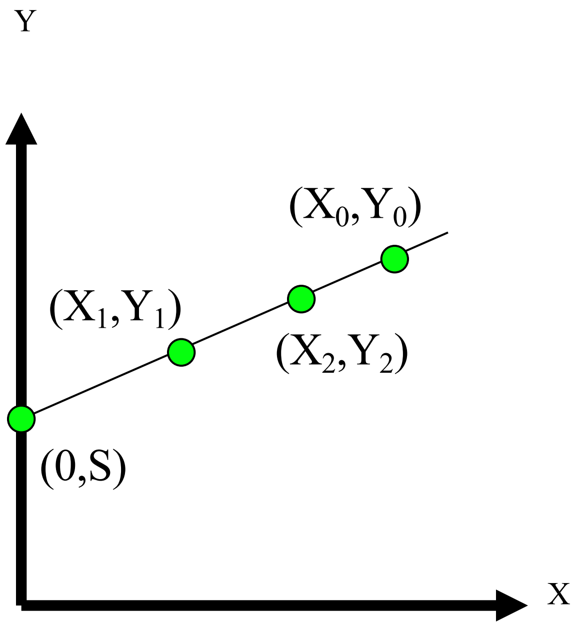


2 out of 2

Two points determine a line

- Give (X_0,Y_0) to Alice
- Give (X_1,Y_1) to Bob
- Then Alice and Bob must cooperate to find secret S
- Also works in discrete case

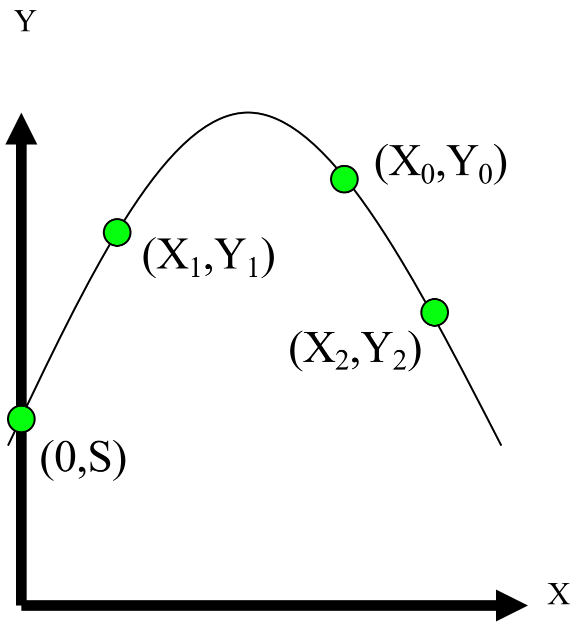
Shamir's Secret Sharing



2 out of 3

- Give (X_0,Y_0) to Alice
- Give (X_1,Y_1) to Bob
- Give (X_2,Y_2) to Charlie
- Then any two can cooperate to find secret S
- But one can't find secret S
- A "2 out of 3" scheme

Shamir's Secret Sharing



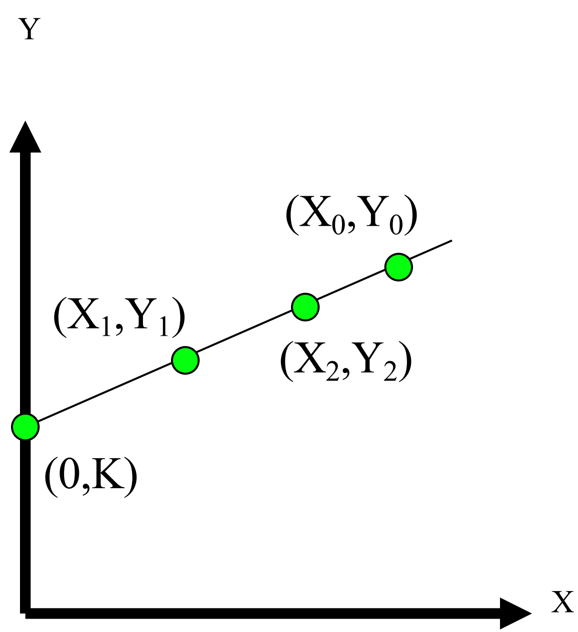
3 out of 3

- Give (X_0,Y_0) to Alice
- Give (X_1,Y_1) to Bob
- Give (X_2,Y_2) to Charlie
- 3 pts determine parabola
- Alice, Bob, **and** Charlie must cooperate to find S
- A "3 out of 3" scheme
- What about "3 out of 4"?

Secret Sharing Example

- **Key escrow** — suppose it's required that your key be stored somewhere
- Key can be "recovered" with court order
- But you don't trust FBI to store your keys
- We can use secret sharing
  - Say, three different government agencies
  - Two must cooperate to recover the key

Secret Sharing Example



Your symmetric key is K

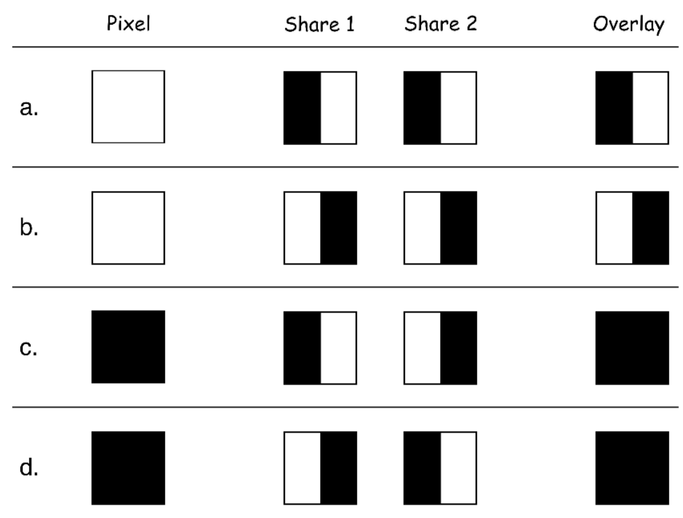
- Point (X_0,Y_0) to FBI
- Point (X_1,Y_1) to DoJ
- Point (X_2,Y_2) to DoC
- To recover your key K, two of the three agencies must cooperate
- No one agency can get K

Visual Cryptography

- Another form of secret sharing…
- Alice and Bob “share” an image
- Both must cooperate to reveal the image
- Nobody can learn anything about image from Alice’s share or Bob’s share
  - That is, both shares are required
- Is this possible?

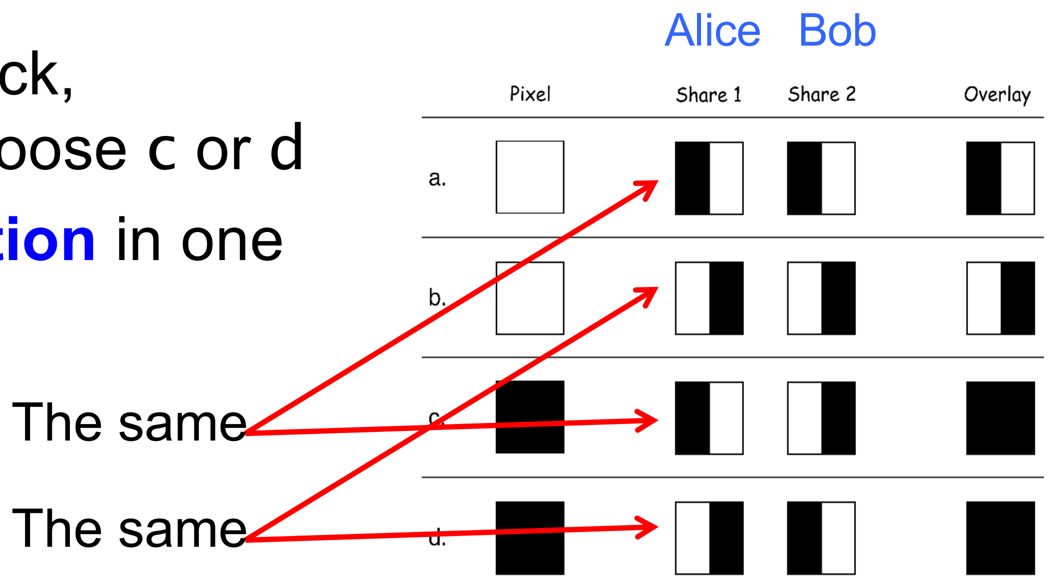
Visual Cryptography

- How to share a pixel?
- Suppose image is black and white
  - Then each pixel is either black or white
  - We split pixels as shown



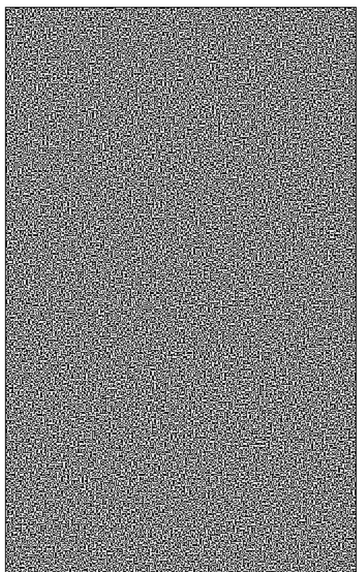
Sharing a B&W Image

- If pixel is white, randomly choose a or b for Alice's/Bob's shares
- If pixel is black, randomly choose c or d
- **No information** in one "share"

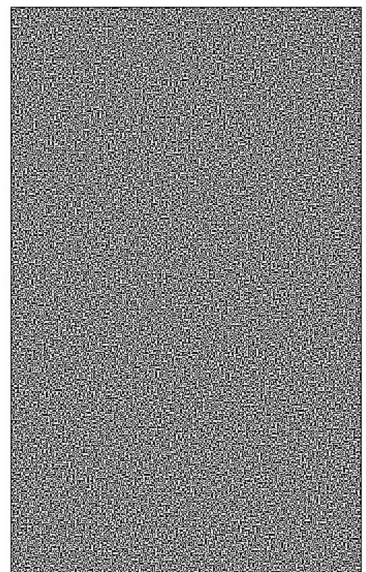


Visual Crypto Example

- Alice's share



- Bob's share



- Overlaid shares

BINGHAMTON UNIVERSITY
STATE UNIVERSITY OF NEW YORK

Visual Crypto

- How does visual “crypto” compare to regular crypto?
- In visual crypto, no key…
  - Or, maybe both images are the key?
- With encryption, exhaustive search
- Exhaustive search on visual crypto?
  - No exhaustive search is possible!

Information Hiding

Information Hiding

- Digital Watermarks
  - Example: Add “invisible” identifier to data
  - Used to prove ownership of the data
  - Defense against music or software piracy

- Steganography
  - *Greek: Steganos (*"covered, concealed, or protected”) + *graphein (*“writing”*)*
  - “Secret” communication channel
  - Example: Hide data in image or music file

BINGHAMTON
UNIVERSITY
STATE UNIVERSITY OF NEW YORK

Watermark

- Add a “mark” to data
- Visibility of watermarks
  - Invisible — Watermark is not obvious
  - Visible — Such as **TOP SECRET**
- Robustness of watermarks
  - Robust — Readable even if attacked
  - Fragile — Damaged if tampered

Watermark Example (1)

- Non-digital watermark: U.S. currency

Where is the watermark?

Andrew Jackson: 7th US president

Watermark Example (2)

- Add **invisible** watermark to photo
- Claimed that 1 $inch^2$ contains enough info to reconstruct entire photo
- If photo is damaged, watermark can be used to reconstruct it!

Steganography Example

- Example: according to Herodotus (A Greek historian 484-425 BC)
  - A Greek general shaved slave's head
  - Wrote message on head
  - Let hair grow back
  - Sent slave to deliver message
  - Another Greek general shaved slave's head to expose message — warning of Persian invasion
- Historically, steganography used more often than cryptography

Images and Steganography

- Images use 24 bits for color: **RGB**
  - 8 bits for red, 8 for green, 8 for blue
- For example
  - **0x7E 0x52 0x90** is this color
  - **0xFE 0x52 0x90** is this color
- While
  - **0xAB 0x33 0xF0** is this color
  - **0xAB 0x33 0xF1** is this color
- Low-order bits don't matter…

Images and Stego

- Given an uncompressed image file…
- …we can insert information into low-order RGB bits
- Since low-order RGB bits don't matter, result will be "invisible" to human eye
  - But, computer program can "see" the bits

Stego Example 1: what's the difference?

- Left side: plain Alice image
- Right side: Alice with entire *Alice in Wonderland* (pdf) "hidden" in the image

Stego Example

- Walrus.html in web browser

"The time has come," the Walrus said,
"To talk of many things:
Of shoes and ships and sealing wax
Of cabbages and kings
And why the sea is boiling hot
And whether pigs have wings."

- "View source" reveals:

```
<font color=#000000>"The time has come," the Walrus said,</font><br>
<font color=#000000>"To talk of many things: </font><br>
<font color=#000000>Of shoes and ships and sealing wax </font><br>
<font color=#000000>Of cabbages and kings </font><br>
<font color=#000000>And why the sea is boiling hot </font><br>
<font color=#000000>And whether pigs have wings." </font><br>
```

Stego Example 2

- stegoWalrus.html in web browser

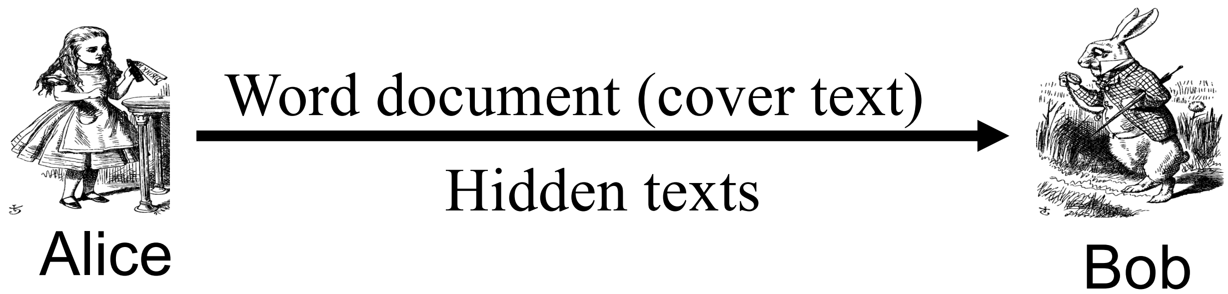
"The time has come," the Walrus said,
"To talk of many things:
Of shoes and ships and sealing wax
Of cabbages and kings
And why the sea is boiling hot
And whether pigs have wings."

- "View source" reveals:

```
<font color=#000101>"The time has come," the Walrus said,</font><br>
<font color=#000100>"To talk of many things: </font><br>
<font color=#010000>Of shoes and ships and sealing wax </font><br>
<font color=#010000>Of cabbages and kings </font><br>
<font color=#000000>And why the sea is boiling hot </font><br>
<font color=#010001>And whether pigs have wings." </font><br>
```

- "Hidden" message: **011 010 100 100 000 101**

Another steganography example



- Alice and Bob share a secret key K
- Alice wants to send a hidden text **C** to Bob
- For every sentence in the doc, Alice calculates its keyed hash **H**
  - If **C = H**, use that sentence;

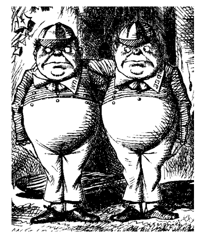
Robustness of steganography

Alice

Cover text

Hidden text

Bob



Trudy uses active attacks

- **Robustness** means if Trudy uses active attacks by modifying the messages in between, can Alice and Bob still communicate through the secret channel

Information Hiding: The Bottom Line

- Not-so-easy to hide digital information
  - "Obvious" approach such as LSB steganography is **not** robust
  - Stego/watermarking active research topics
- If information hiding is suspected
  - Attacker may be able to make information/watermark unreadable
  - Attacker may be able to read the information, given the original document (image, audio, etc.)

Summary of Cryptography

- **Basics of cryptology**
  - Crypto terms
  - Kerckhoffs' Principle
  - Shift crypto, double transposition, one-time pad, codebook cipher
- **Symmetric key crypto**
  - Stream cipher: A5/1, RC4
  - Block cipher: Feistel cipher, DES, 3DES, AES, IDEA, Blowfish, RC6
  - Block cipher modes: ECB, CBC, CTR
  - MAC
- **Public key crypto**
  - Knapsack cryptosystem (flawed), RSA, Diffie-Hellman, El Gamal, ECC
  - Uses for public key crypto, PKI
- **Cryptographic Hash functions**
  - Birthday problem
  - Tiger hash
  - HMAC