HMAC

- Can compute a MAC of the message M with key K using a "hashed MAC" or **HMAC**
- HMAC is a ***keyed*** hash
  - Why would we need a key?
- How to compute HMAC?
- Two obvious choices: h(K,M) and h(M,K)
- Which is better?

HMAC

- Should we compute HMAC as $h(K,M)$?
- Hashes computed in blocks
  - $h(B_1,B_2) = F(F(A,B_1),B_2)$ for some F and constant A
  - Then $h(B_1,B_2) = F(h(B_1),B_2)$
- Let $M' = (M,X)$
  - Then $h(K,M') = F(h(K,M),X)$
  - Attacker can compute HMAC of M' without K
- Is $h(M,K)$ better?
  - Yes, but… if $h(M') = h(M)$ then we might have $h(M,K)=F(h(M),K)=F(h(M'),K)=h(M',K)$

Correct Way to HMAC

- Let B be the block length of hash, in bytes
  - B = 64 for MD5 and SHA-1 and Tiger
- ipad = 0x36 repeated B times
- opad = 0x5C repeated B times
- Then

$$\text{HMAC}(M,K) = h(K \oplus \text{opad}, h(K \oplus \text{ipad}, M))$$

Hash Uses

- Authentication (HMAC)
- Message integrity (HMAC)
- Message fingerprint
- Data corruption detection
- Digital signature efficiency
- Anything you can do with symmetric crypto
- Also, many, many clever/surprising uses…

Online Bids

- Suppose Alice, Bob and Charlie are bidders
- Alice plans to bid A, Bob B and Charlie C
- They don't trust that bids will stay secret
- A possible solution?
  - Alice, Bob, Charlie submit **hashes** h(A), h(B), h(C)
  - All hashes received and posted online
  - Then bids A, B, and C submitted and revealed
- Hashes don't reveal bids (one way)
- Can't change bid after hash sent (collision)

Hashing for Spam Reduction

- Spam reduction
- Before accept email, want proof that sender had to "work" to create email
  - Here, "work" == CPU cycles
- Goal is to limit the amount of email that can be sent
  - This approach will not eliminate spam
  - Instead, make spam more costly to send

Spam Reduction

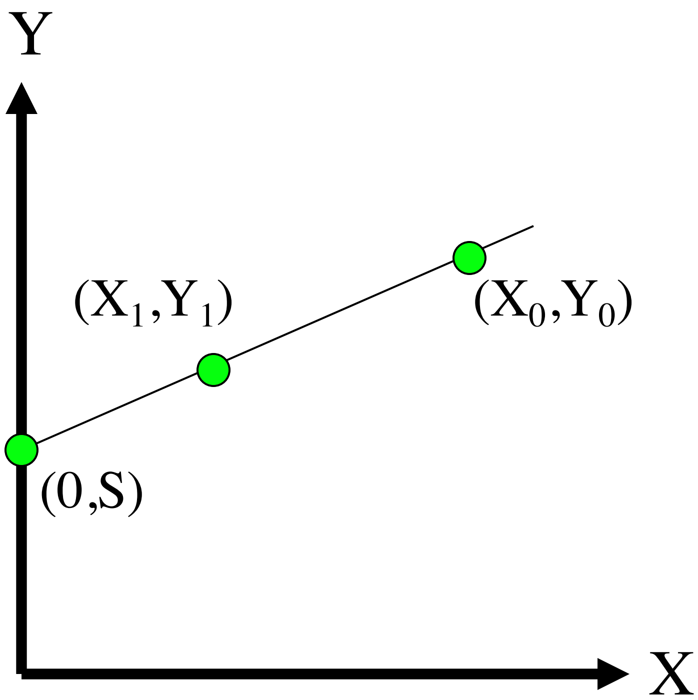
- Let M = complete email message
  R = value to be determined
  T = current time
- Sender must determine R so that
  h(M,R,T) = (00…0,X), that is,
  initial N bits of hash value are **all zero**
- Sender then sends (M,R,T)
- Recipient accepts email, provided that…
  h(M,R,T) begins with N zeros

Spam Reduction

- Sender: $h(M,R,T)$ begins with N zeros
- Recipient: verify that $h(M,R,T)$ begins with N zeros
- **Work for sender:** on average 2^N **hashes**
- **Work for recipient:** always 1 **hash**
- Sender's work increases exponentially in N
- Small work for recipient, regardless of N
- Choose N so that…
  - Work acceptable for normal amounts of email
  - Work is too high for spammers

Secret Sharing

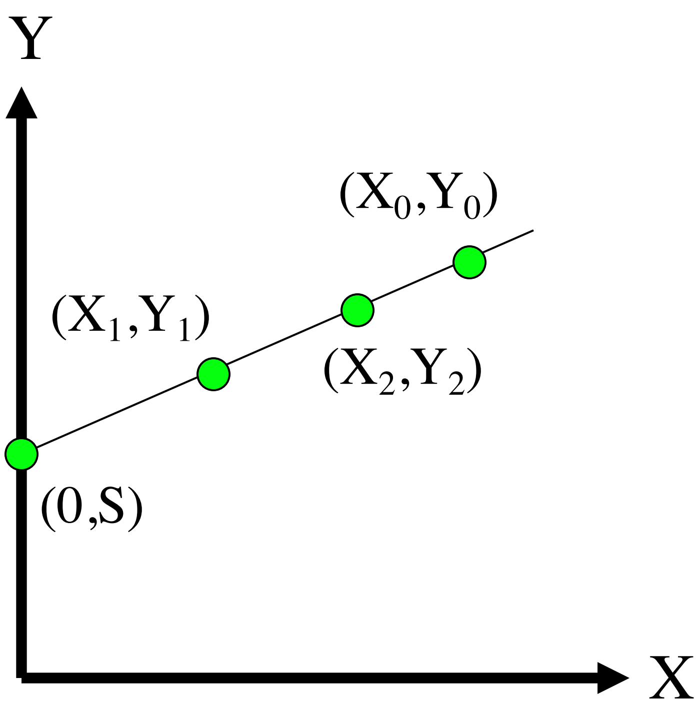
Shamir's Secret Sharing



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
- Easy to make "m out of n" scheme for any $m \leq n$

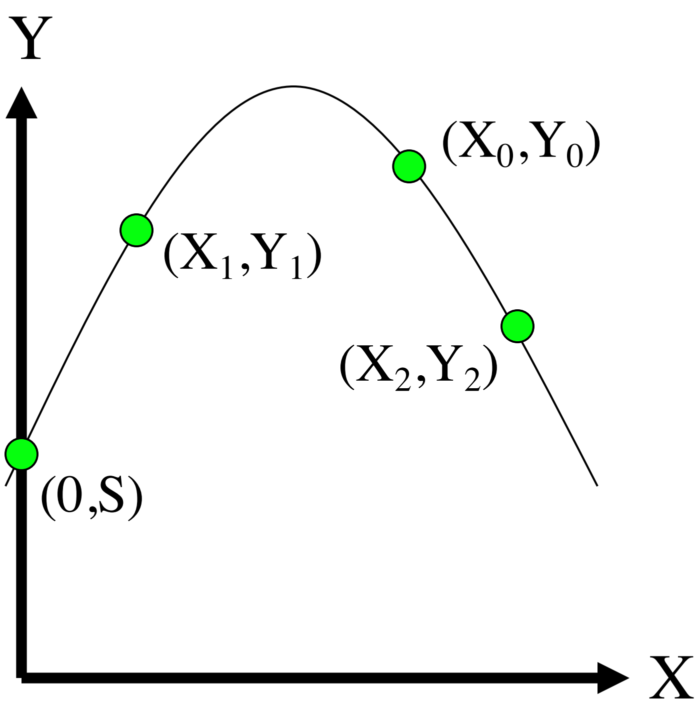
Shamir's Secret Sharing



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
- No *one* can determine S
- A "2 out of 3" scheme

Shamir's Secret Sharing



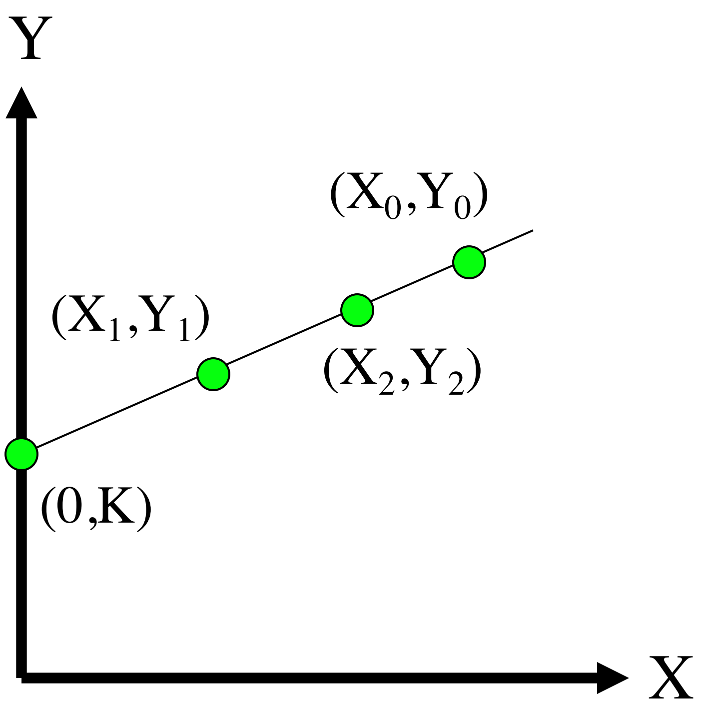
3 out of 3

- 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 Use?

- **Key escrow** — suppose it's required that your key be stored somewhere
- 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



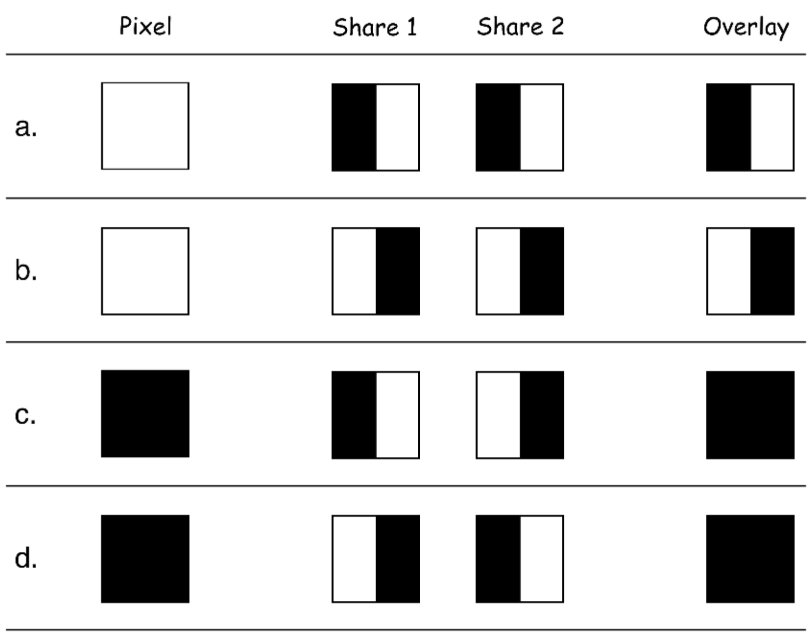
- Your symmetric key is K
- Point (X_0,Y_0) to FBI
- Point (X_1,Y_1) to DoJ
- Point (X_2,Y_2) to DoC
- To recover your key K, two of the three agencies must cooperate
- No one agency can get K

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?

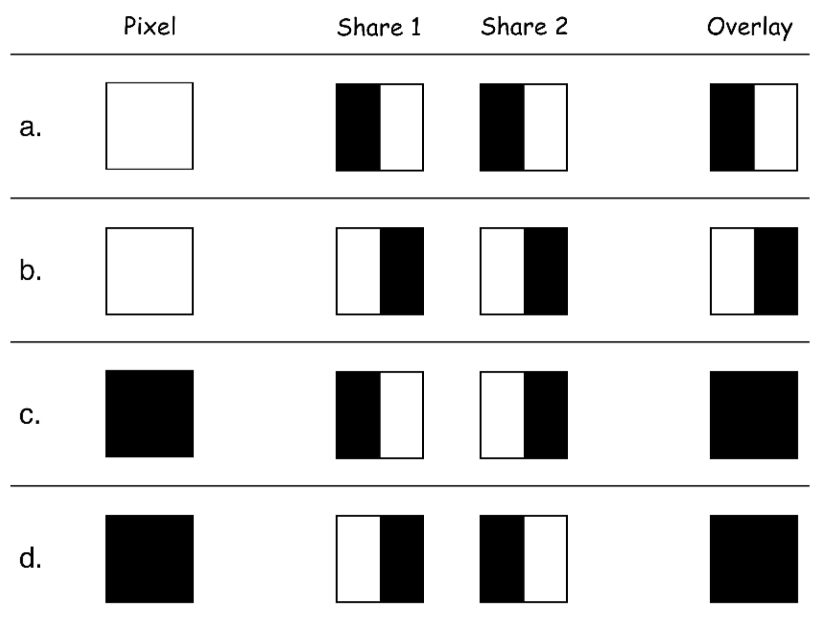
Visual Cryptography

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



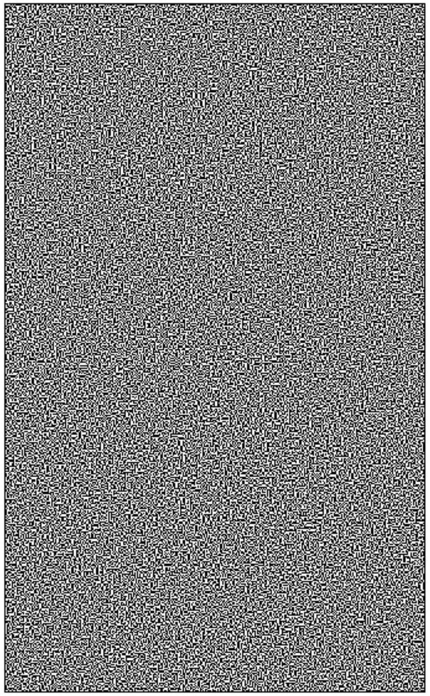
Sharing Black & White Image

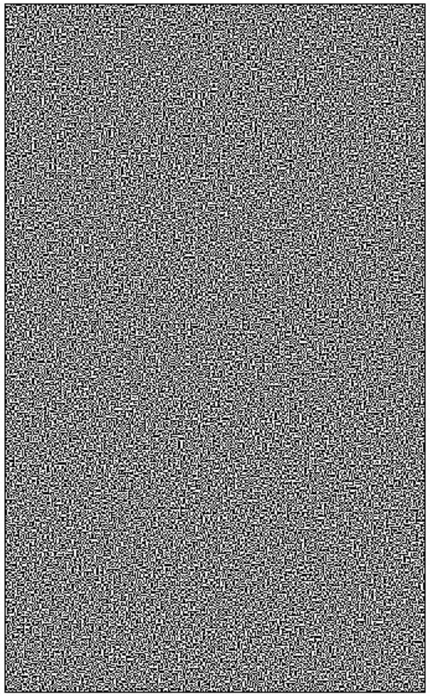
- If pixel is white, randomly choose a or b for Alice's/Bob's shares
- If pixel is black, randomly choose c or d
- **No information** in one "share"



Visual Crypto Example

- Alice's share
- Bob's share
- Overlaid shares





Random Numbers in Cryptography

Random Numbers

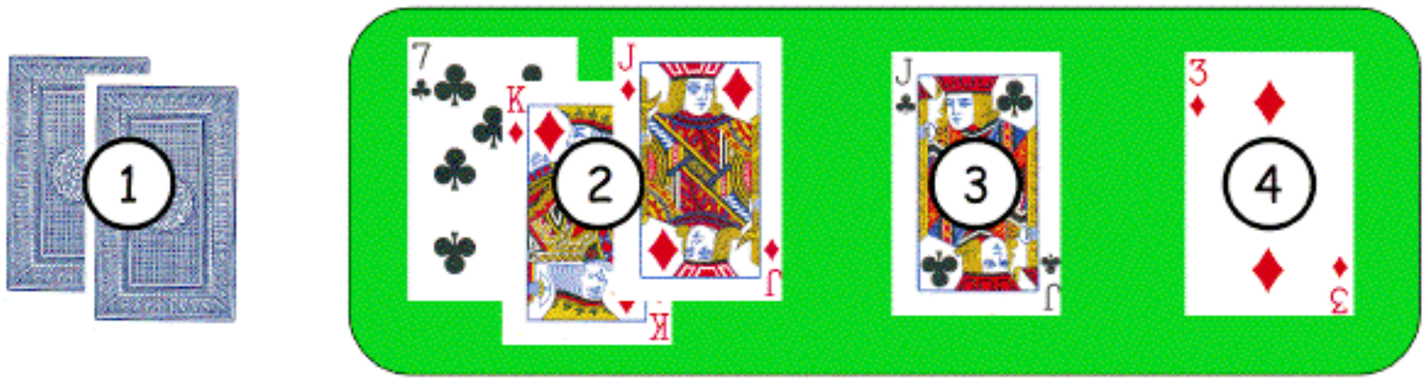
- Random numbers used to generate **keys**
  - Symmetric keys
  - RSA: Prime numbers
  - Diffie Hellman: secret values
- Random numbers also used in simulations, statistics, etc.
  - "statistically" random numbers

Random Numbers

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



Player's hand — Community cards in center of the table

- 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 Pascal pseudo-random number generator (PRNG): Randomize()
- Seed value for PRNG was function of number of milliseconds since midnight
- Less than 2^{27} milliseconds in a day
  - So, less than 2^{27} possible shuffles

Card Shuffle

- Seed based on milliseconds since midnight
- PRNG re-seeded with each shuffle
- number of shuffles could be reduced to 2^{18} which can be tested in real time
- Attacker knows every card after the first of five rounds of betting!

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 "random" is hard to define
- Entropy is a measure of randomness
- Good sources of "true" randomness
  - Radioactive decay
  - Hardware devices (e.g. thermal noise, photoelectric effect)
  - Lava lamp — relies on chaotic behavior

Randomness

- Sources of randomness via software
  - Software is supposed to be deterministic
  - So, must rely on external "random" events
  - Mouse movements, keyboard dynamics, network activity, etc.
- Can get quality random bits by such methods
- Bottom line: "The use of pseudo-random processes to generate secret quantities can result in pseudo-security"

Information Hiding

Information Hiding

- Digital Watermarks
  - Example: Add "invisible" info to data
  - Defense against music/software piracy
- Steganography
  - "Secret" communication channel
  - Example: Hide data in an image file

Watermark

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

Watermark Examples

- Add robust invisible mark to digital music
  - If pirated music appears on Internet, can trace it back to original source of the leak
- Add fragile invisible mark to audio file
  - If watermark is unreadable, recipient knows that audio has been tampered with (integrity)
- Combinations of several types are sometimes used
  - E.g., visible plus robust invisible watermarks

Watermark Example

- Non-digital watermark: U.S. currency

- Image embedded in paper
  - Hold bill to light to see embedded info

Steganography

- According to Herodotus (Greece 440 BC)
  - Shaved slave's head
  - Wrote message on head
  - Let hair grow back
  - Send slave to deliver message
  - Shave slave's head to expose a message warning of Persian invasion
- Historically, steganography used by military 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…
  - For example, BMP format
- …we can insert information into low-order RGB bits
- Since low-order RGB bits don't matter, changes will be "invisible" to human eye
  - But, computer program can "see" the bits

Watermarking Example

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

Steganography

- Some formats (e.g., image files) are more difficult than html for **humans** to read
  - But easy for computer programs to read…
- Easy to hide info in **unimportant bits**
- Easy to damage info in unimportant bits
- To be ***robust***, must use **important bits**
  - But stored info must not damage data
  - Collusion attacks are also a concern
- Robust steganography is tricky!

Information Hiding: The Bottom Line

- Not-so-easy to hide digital information
  - "Obvious" approach is **not** robust
  - Stego/watermarking are 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.)