Man vs. Machine (human-computer cryptography)

Dr. István Zsolt Berta
www.berta.hu
istvan@berta.hu

opinions expressed here
are strictly those of my own

Human-computer cryptography

- Strong cryptographic algorithms are complex, we use computers for cryptographic operations
- ‘Human-computer cryptography’ or ‘pencil-and-paper cryptography’ deals with algorithms executable by humans
- Can a human encrypt/authenticate messages without a computer, with a security that can help against today’s attackers?

Why human-computer crypto?

- Useful if no computer is available
- Useful if no **trusted** computer is available
  (i.e. you have a computer but do not trust it)
- Out-of-the-box thinking about cryptography

- It is fun! 

Man vs Machine

- Performs operations slowly
- Can memorize secrets of a few characters only
- Makes mistakes
- Capable of feelings: love, empathy, etc.

- Performs operations quickly
- Has gigabytes of memory and secure storage
- Almost no mistakes
- No feelings, just silicon

Disclaimers

- I shall not show you a quick & easy & comfortable way for a human to do secure encryption without computers → if there was such a way, we would not be bothering with machines
- I am not going speak about anything new, I shall cite publicly available research papers
- Many solutions will be simple if not obvious; in fact, complex things do not work, humans cannot execute them properly
- I shall not go into math details, I shall provide an overview
- I do not claim to have a complete overview on the topic; feel free to share any other solutions / ideas you are aware of

What does NOT work

- Most cryptography from before the age of computers is useless against today's attackers
  - scytale, Caesar cipher, Vigenère cipher, etc. offer but little more protection than rot13
  - vulnerable to frequency analysis and/or
  - can be brute forced easily

- Their slightly tweaked versions are equally useless
- Performing modern crypto algorithms (e.g. RSA, AES or 3DES) with pencil and paper?
  → too hard, better forget it
  (though RC4 may be an option for *very* motivated humans)

Human-computer encryption

One-time-pad

- For encrypting *n* bits of plaintext, *n* bits of key are needed; ciphertext can be obtained by XOR-ing the bits of the plaintext and the key
- Example:

	01000111011001	plaintext input
	10110111011101	key input
⊕	11110000000100	ciphertext output

- Note that the key must be truly random, and keybits shall **never** be reused; otherwise encryption becomes very weak
- Very hard key management problem, one-time-pads are almost never used properly in practice

One-time-pad: perfect secrecy

- OTP provides perfect, unconditional secrecy
- If used properly, encryption is secure, regardless of the resources and capabilities of the attacker
- Plaintext and ciphertext are independent random variables; no way to deduce plaintext from ciphertext
- A brute force search of all possible keys reveals all possible messages with equal probability
- Proof: Shannon, 1949

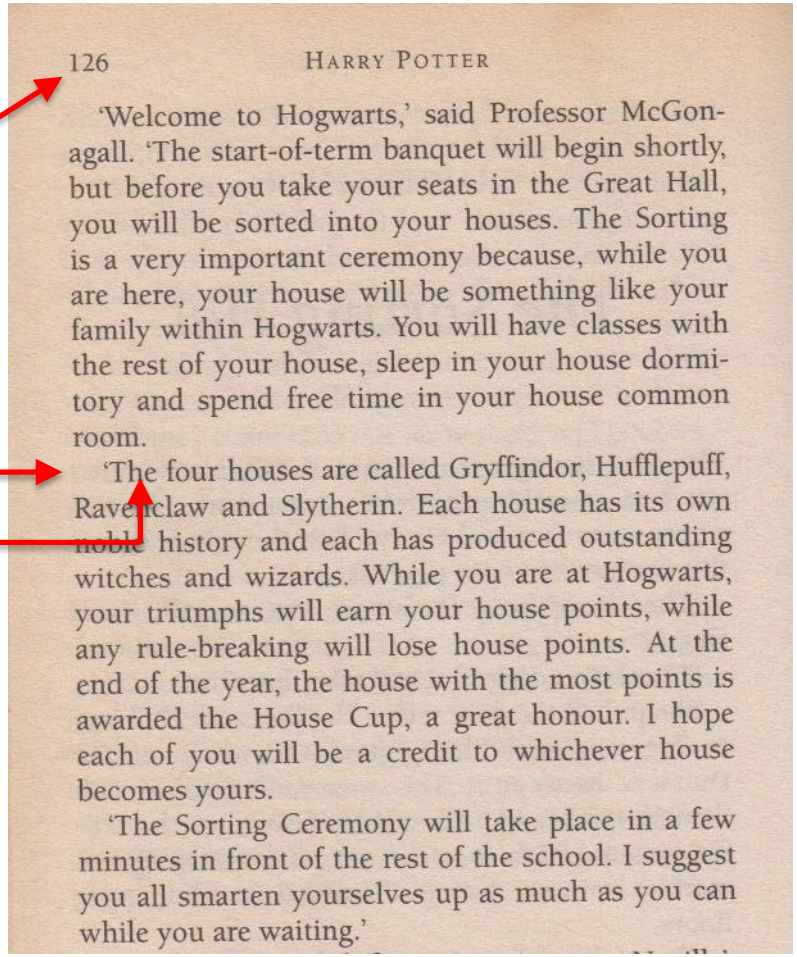
- However, *n* non-reusable random bits need to be transferred securely to transfer *n* bits securely…

One-time-pad: Human vs computer?

- Simple algorithm but a human cannot memorize long one-time keys
- Usable for *very* short messages (~few words)
- Usable if the human can store the keys securely; e.g. keys stored on paper are out-of-reach for online attackers
- Addition should not be mod2, but a character-wise (e.g. mod26) operation, this can be aided with public tables

Book cipher

- Historical cipher that can still work if used properly
- A book is used as a key, ciphertext contains offsets of characters in the book
- E.g. 126-11-2 = “h” (page 126, line 11, char 2)
- Books are used as an easy way for transporting keys
  - a book is not suspicious
  - if recipient can buy the same book, no need to transport it



126 HARRY POTTER

‘Welcome to Hogwarts,’ said Professor McGonagall. ‘The start-of-term banquet will begin shortly, but before you take your seats in the Great Hall, you will be sorted into your houses. The Sorting is a very important ceremony because, while you are here, your house will be something like your family within Hogwarts. You will have classes with the rest of your house, sleep in your house dormitory and spend free time in your house common room.

‘The four houses are called Gryffindor, Hufflepuff, Ravenclaw and Slytherin. Each house has its own noble history and each has produced outstanding witches and wizards. While you are at Hogwarts, your triumphs will earn your house points, while any rule-breaking will lose house points. At the end of the year, the house with the most points is awarded the House Cup, a great honour. I hope each of you will be a credit to whichever house becomes yours.

‘The Sorting Ceremony will take place in a few minutes in front of the rest of the school. I suggest you all smarten yourselves up as much as you can while you are waiting.’

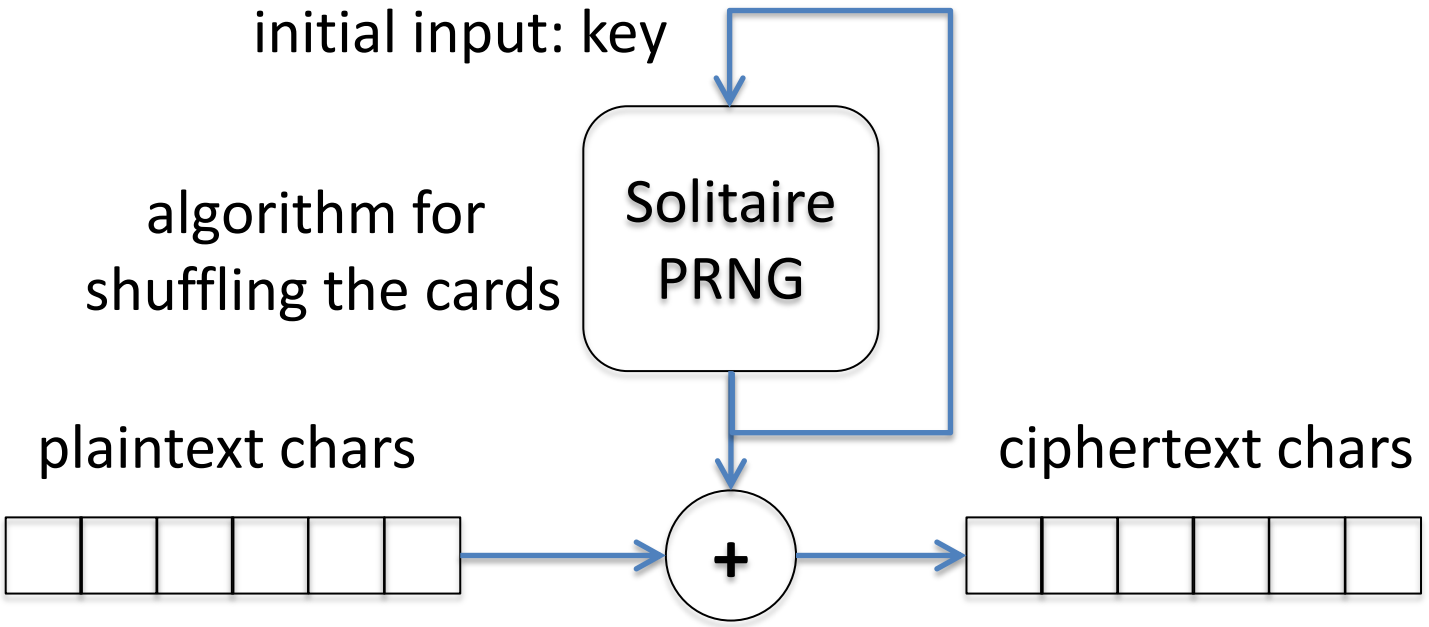
Book Cipher vs Computers

- Always pick different locations for the same character, never reuse locations (to avoid making it a mono-alphabetic cipher)
- Do not pick characters close to each other, they can have some correlation
- Most importantly: **PROTECT THE BOOK!**
- If the attacker finds out which book you use, the cipher becomes useless
- Note: Today it is realistic to perform a brute force search of all books/writings/etc. ever published
- Unpublished writings could provide good encryption, but they need to be transported to the recipient

Solitaire

- A cipher developed by Bruce Schneier, it appears in the novel Cryptonomicon as ‘Pontifex’
- Solitaire is an output-feedback stream cipher, it defines a systematic method for shuffling a deck of cards, acting as a pseudo-random number generator (PRNG)
- Plaintext is added to the output of the PRNG mod26
- The initial ordering of cards is the key
- A deck of cards is not suspicious to have; the key can be destroyed by shuffling the deck randomly
- Details: https://www.schneier.com/solitaire.html

Solitaire: output-feedback stream cipher



- Such stream ciphers are imperfect one-time-pads
- The same algorithm is used for decryption; recipients adds ciphertext to the same random numbers mod26 to obtain plaintext

Solitaire: shuffling algorithm

For each plaintext letter:

1) Find the A joker. Move it one card down.
2) Find the B joker. Move it two cards down.
3) Perform a triple cut. That is, swap the cards above the first joker with the cards below the second joker.
4) Perform a count cut. Look at the bottom card. Convert it into a number from 1 through 53. [...] Count down from the top card that number. [...] Cut after the card that you counted down to, leaving the bottom card on the bottom.
5) Find the output card. To do this, look at the top card. Convert it into a number from 1 through 53 in the same manner as step 4. Count down that many cards. [The output card is the next one.]
6) Convert the output card to a number [, add it to the plaintext number.]

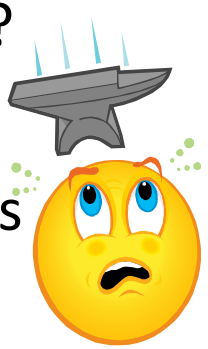
Solitaire: Security

- Solitaire was designed against the most well-funded military adversaries with the biggest computers and the smartest cryptanalysts
- Successful cryptanalysis: A bias was found in Solitaire's PRNG, i.e. certain random numbers are more likely than others
  - http://www.ciphergoth.org/crypto/solitaire/
  - Pogorelov&Pudovkina, 2003

- Solitaire is considered the most serious attempt...

Handycipher

- The key is a 30-character-long permutation of letters in the alphabet (&some symbols)
- Two tiers:
  - a mono-alphabetic substitution
  - another substitution, picked ’randomly’ (based on the key) from a set of different algorithms
- Author’s rationale: So much noise (randomness) is added that – being a pencil-and-paper cipher – it is unlikely the attacker would collect enough data for successful cryptanalysis.... ?
- Research paper: Kallick, 2014
- Very new, no experience & no independent research on its security, no clue how secure... risky option...

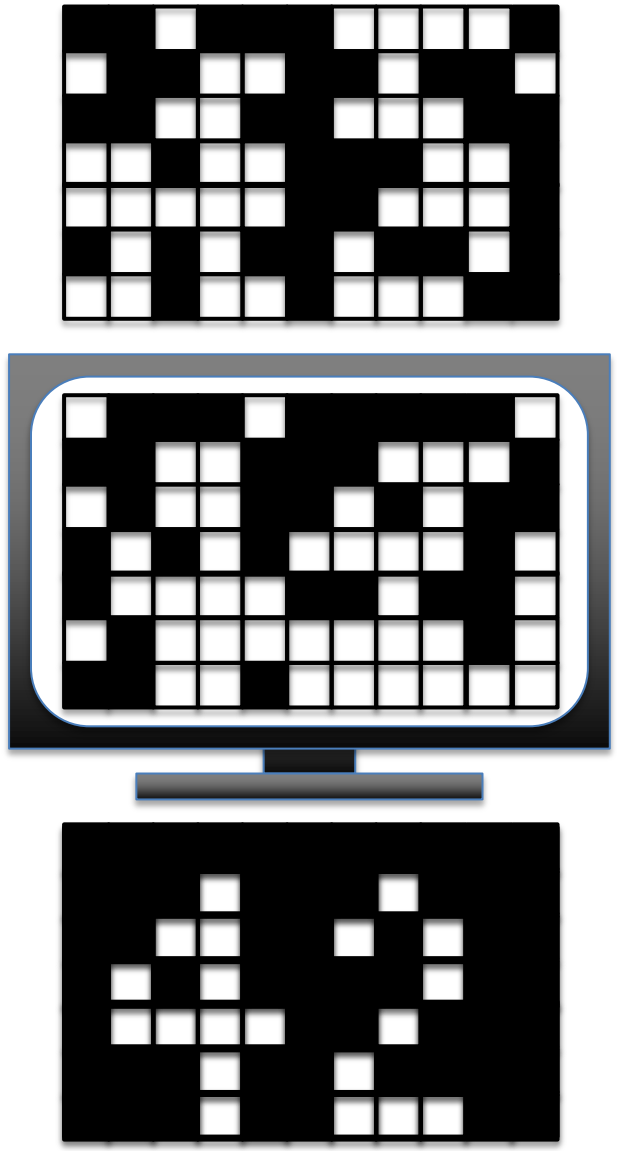


Visual Cryptography

- Used for transferring images
- The human is carrying a set of transparencies, i.e. slides with transparent and non-transparent cells
- The human receives an image, and places a transparency over it, and thus is the message revealed
- One transparency is used for sending one message only, and must not be reused
- Research paper: Shamir&Naor, 1994

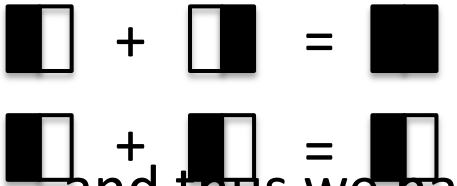
Visual Cryptography: how it works

- The user carries a transparency
- She receives an image (e.g. on the screen of her untrusted computer)
- She places the transparency onto the image on the screen...
- and she sees the encrypted message

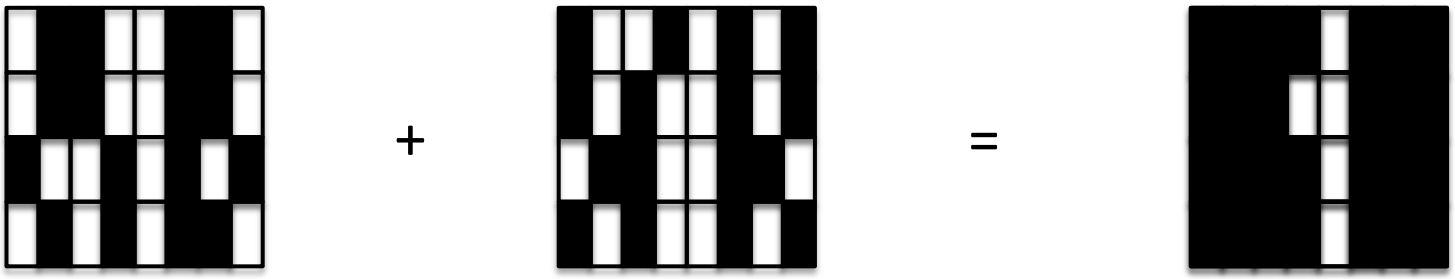


Visual Cryptography: let's make it more secure!

- By now, we were able to hide white cells only, and this is weak as the attacker knows that all black cells will remain black
- Let's define cells a different way: half of the cell is alway black, the other half is always white (transparent)



... and thus we have an XOR operation!



Visual Cryptography: security

- This is a one-time-pad, where the XOR operation is accelerated by the human eye
- Perfect, unconditional secrecy, etc.

- **Thou shalt not reuse thy transparencies!**
  (otherwise the encryption becomes very weak)

- Key management is a problem, transparencies must be transferred to the recipient in a secure manner

Visual Cryptography: tweaks

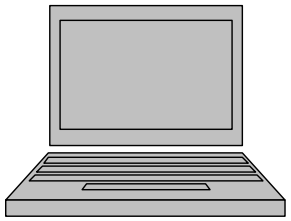
- Can be generalized as a secret sharing scheme, where (transparencies from) *k* people are needed to reveal an image
- There are extensions for transferring e.g. grayscale images (via high-resolution transparencies, where the human eye interprets black and white dots close to each other as gray)
- Can be modified for message authentication, research paper: Naor&Pinkas, 1995

Message via different channels

- Assume we have two different computers (e.g. a laptop and a smartphone); we trust neither of them, but assume that they do not cooperate against us
- A simple, one-time-pad solution can be used:

- send random bits on one channel,

- send message $\oplus$ same random bits on the other channel



How about hiding the message?

Cryptography

- The attacker knows all details of the system, except for the private/secret keys (Kerckhoffs's principle)
- The attacker is omnipresent, it can intercept (and possibly modify) all messages
- The attacker obtains the ciphertext

Steganography

- The attacker does not know exactly how the message is hidden
- The attacker needs to select the message from a lot of harmless messages
- If the attacker selects the message, the game is over

They are often used in combination (first encrypt, then hide). This not only provides an additional layer of security, encryption hides the structure of the message, making the hidden message harder to spot.

Simple steganography

- There are very simple ways for hiding messages, which are very hard to detect automatically
- Hiding messages in lower bits of images is hard for humans
- Humans can hide messages in the **content** of e.g. a video message, and these are very hard to detect with machines
- Example: if someone is speaking in the *n*th minute mark, it means a “1”, otherwise it is a “0”

- Note: This is not cryptography, we do not even try to meet Kerckhoffs’s principle

Authentication of an unaided human

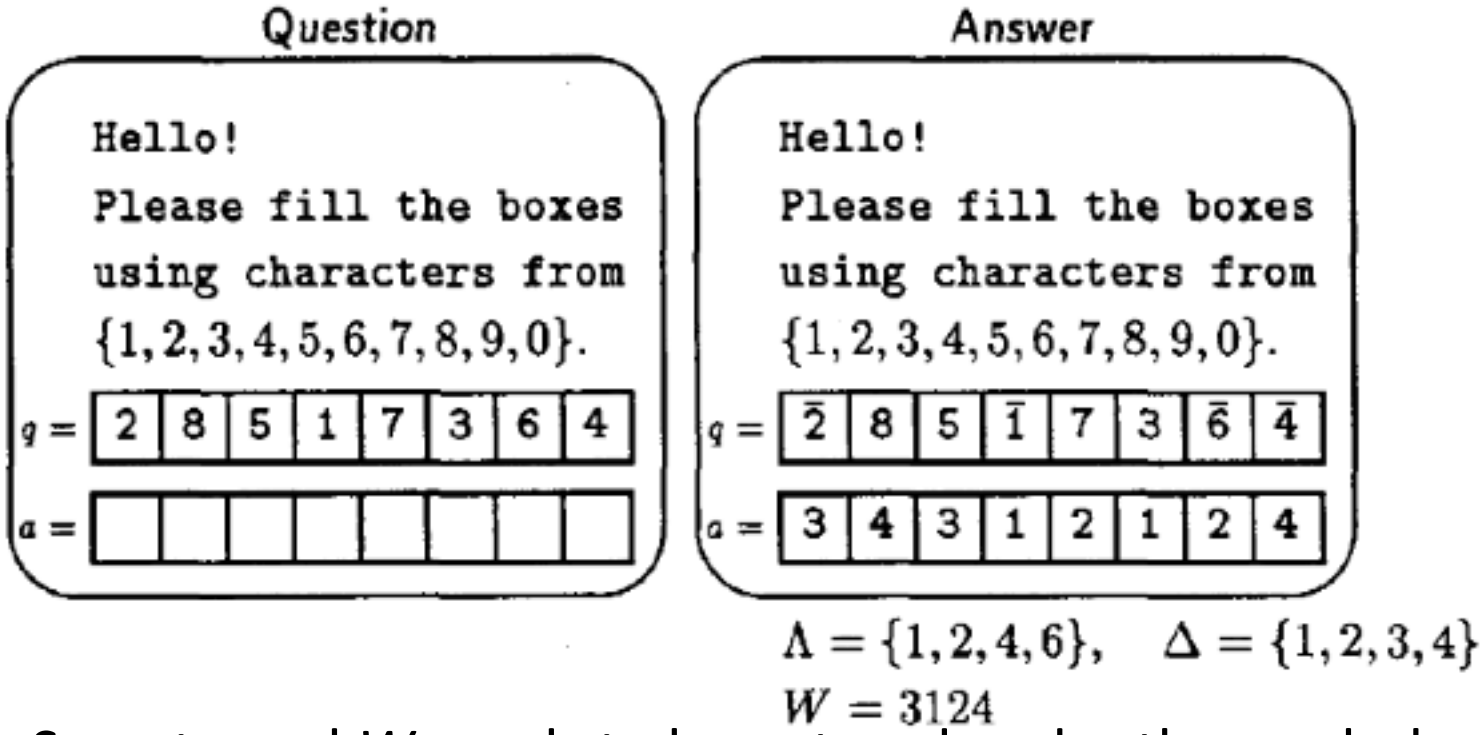
One-time-passwords

- A printed, paper-based list of one-time passwords is out-of-reach for online-only attackers; if sufficiently long passwords are used, this can provide good security
- Example:

1st	ZNoZZ9=JOZGZ…
2nd	UyzjL7l#-0my…
3rd	8iZJKPLJjdH6Vbnp…
…	…

- Passwords must not be reused
- Of course, the human will never be able to memorize a long list, it needs to be on paper

Matsumoto&Imai, 1991



- Secret word *W* needs to be entered under the symbols of Λ; enter chars from Δ under other symbols at random
- Research paper: Matsumoto&Imai, 1991
- Shown to be insecure: Wang&Hwang&Tsai, 1995
- Improvement: Matsumoto, 1996

Hopper&Blum, 2001

- The secret key is a vector, the user receives a challenge as a matrix, and needs to multiply it with the vector to respond
- This would be weak, the attacker could obtain the key via Gaussian elimination, provided that enough challenges and responses are observed
- Trick: The user can give incorrect responses with a certain (low) probability, but she can still be authenticated with multiple challenges
- Because of possible wrong responses, Gaussian elimination does not work, in fact the problem becomes NP-hard
- The user makes mistakes, the solution makes this an advantage

- Research paper: Hopper&Blum, 2001

Biometry…

- Biometry is a rather straightforward way to authenticate humans, it does not require computers
- Most biometric solutions are inherently weak to replay attacks

Human-computer message authentication

One-time solution

- We have two lists of one-time passwords printed on paper, one of the lists is for sending 0s, the other is for sending 1s; passwords are shared with the recipient
- Example:

	"0"	"1"
1st bit:	nZho,=…	89HFT…
2nd bit:	J7mzt>…	89zTJ…
3rd bit:	3ky+ld…	qeQQd…

- For each bit of the message, either the password for "0" or the password for "1" needs to be sent to the recipient
- Passwords must not be reused

Lamport signature

- Same as previous solution (with lots of printed one-time passwords), but the hash of each password is published beforehand
- Thus when sending/publishing a password, everyone can verify which bit the user commits herself to
- Hashes cannot be computer by a user, she needs a computer beforehand for computing them

Using Visual Cryptography

- Certain areas on the message are required to be black, any non-black cells in those areas are signs of someone tampering with the message
- Research paper: Naor&Pinkas, 1995

Trusted device with camera

- The message is prepared using an untrusted device
- The user also has a trusted device with no user interface but a camera; the trusted device can read the message from the computer's screen to verify what is signed
- Research paper: Clarke&…&Rivest, 2002

- Note: In 2002 such a device was not realistic. Today we have smartphones, they have both user interface and camera, but are they trusted?

Signature and biometry

- The user has a PKI smart card with a private key, the card is trusted by the user but has no user interface
- Trick: Instead of a plaintext message, the user sends a biometric (voice or video) message; these combine content and user identity; the biometric message is signed with the smart card
- The biometric message is recorded on an untrusted device; there is a protocol for limiting the amount of time the untrusted device has for tampering with the message
- Examples: The use announces the current time at the beginning and the end of the biometric message, the computer also adds a time mark; these must correspond to each other
- Research paper: Berta&Vajda, 2003

Multiple smart cards

- The user has two trusted smart cards with no user interface
- Signing ‘anything’ with one card, means sending “1”, signing ‘anything’ with the other card, means sending “0”

- The user must sign one and only one message in each time slot; signatures must always be created over PKI timestamps, and must always be protected by another PKI timestamp; the two timestamps must be sufficiently close
- The user can control when signatures are created by removing the cards when she does not wish to sign
- Research paper: Berta, 2006

Summary & Conclusions

Summary & Conclusions

- There are solutions for encryption, message authentication, and user authentication; most solutions presented were
  - one-time-pads or one-time password solutions, or
  - awkward / complex / insecure / questionable solutions
- One-time-pads are generally not preferred, because it is too easy to do key management very wrong
- Perhaps, under the resource constraints of a human user, one-time solutions are still the best option; if used well, they provide a known, strong degree of security
- For the average user, secure devices should be the right way

Thank you very much!

Dr. István Zsolt Berta
www.berta.hu
istvan@berta.hu

Man vs. Machine (human-computer cryptography)

Dr. István Zsolt Berta
www.berta.hu
istvan@berta.hu

opinions expressed here
are strictly those of my own