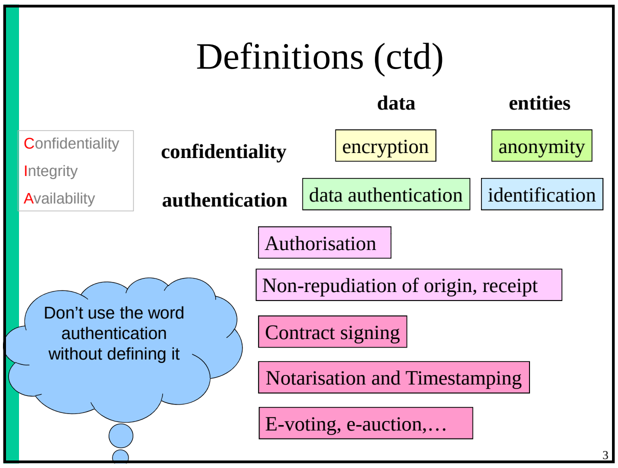
Entity authentication and symmetric key establishment

Prof. Bart Preneel
COSIC
Bart.Preneel(at)esatDOTkuleuven.be
http://homes.esat.kuleuven.be/~preneel

February 2018

Goals

- Understand goals of entity authentication
- Understand strength and limitations of entity authentication protocols including passwords
- Understand subtle problems when entity authentication protocols are deployed in practice
- Understand variants of key establishment protocols and subtle attacks



Definitions (ctd)

Confidentiality
Integrity
Availability

	data	entities
confidentiality	encryption	anonymity
authentication	data authentication	identification

Authorisation

Non-repudiation of origin, receipt

Contract signing

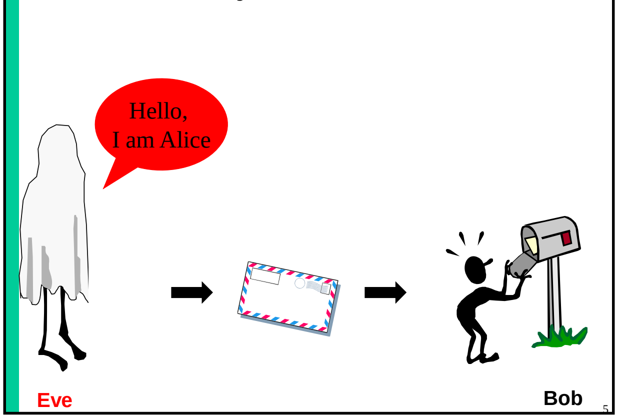
Notarisation and Timestamping

E-voting, e-auction,…

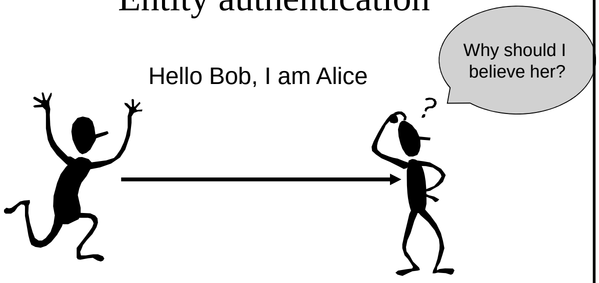
Identification

- the problem
- passwords
- challenge response with symmetric key and MAC (symmetric tokens)
- challenge response with public key (signatures, ZK)
- biometry

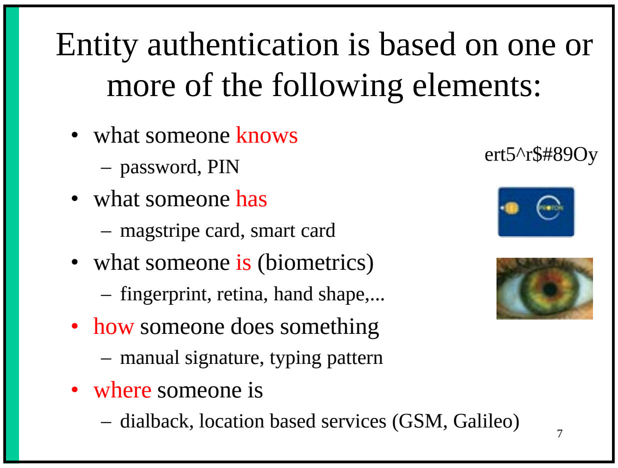
Entity authentication



Entity authentication

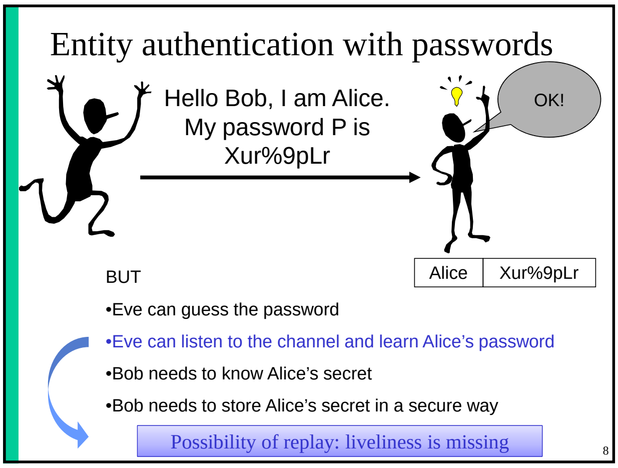


entity authentication: one is corroborated of the identity of another party, and of the fact that this party is **alive (active)** during the protocol



Entity authentication is based on one or more of the following elements:

- what someone knows
  - password, PIN
- what someone has
  - magstripe card, smart card
- what someone is (biometrics)
  - fingerprint, retina, hand shape,...
- how someone does something
  - manual signature, typing pattern
- where someone is
  - dialback, location based services (GSM, Galileo)



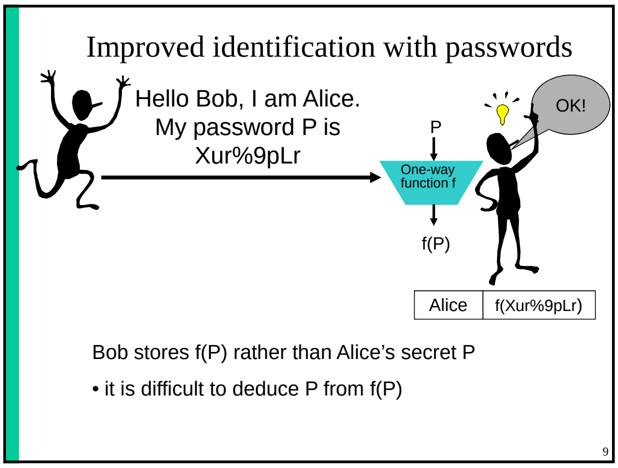
Entity authentication with passwords

Alice	Xur%9pLr

BUT

- Eve can guess the password
- Eve can listen to the channel and learn Alice's password
- Bob needs to know Alice's secret
- Bob needs to store Alice's secret in a secure way

Possibility of replay: liveliness is missing

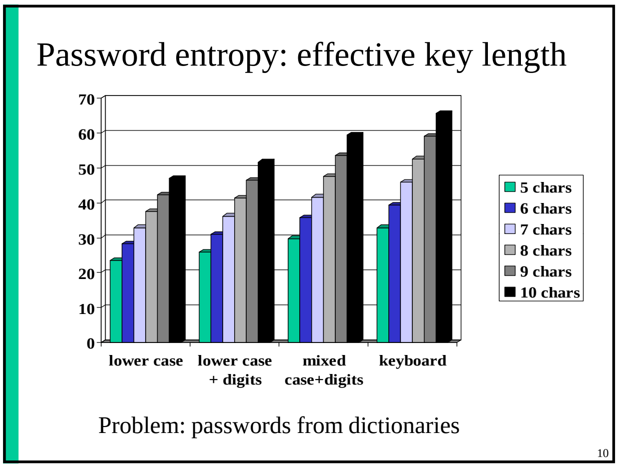


Improved identification with passwords

Alice	f(Xur%9pLr)

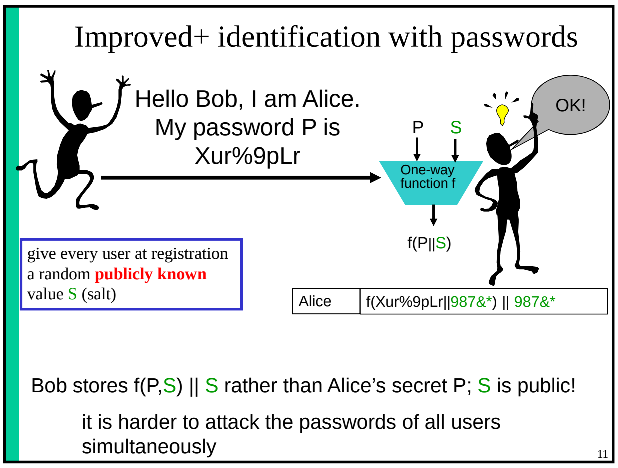
Bob stores f(P) rather than Alice's secret P

- it is difficult to deduce P from f(P)



Password entropy: effective key length

Problem: passwords from dictionaries



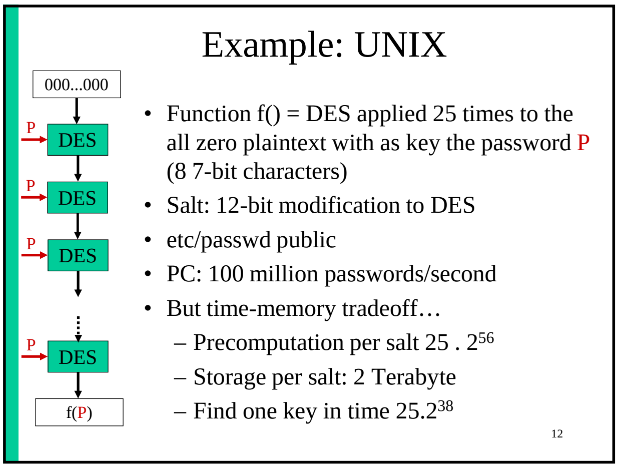
Improved+ identification with passwords

give every user at registration a random **publicly known** value S (salt)

Alice	f(Xur%9pLr||987&*) || 987&*

Bob stores f(P,S) || S rather than Alice's secret P; S is public!

it is harder to attack the passwords of all users simultaneously



Example: UNIX

- Function f() = DES applied 25 times to the all zero plaintext with as key the password P (8 7-bit characters)
- Salt: 12-bit modification to DES
- etc/passwd public
- PC: 100 million passwords/second
- But time-memory tradeoff…
  - Precomputation per salt $25 . 2^{56}$
  - Storage per salt: 2 Terabyte
  - Find one key in time 25.2^{38}

Improving password security

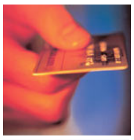
- Apply the function f "x" times to the password (iteratively)
  - if x = 100 million, testing a password guess takes a few seconds
  - need to increase x with time (Moore's law)
  - need to define function f such that special hardware crackers do not gain a large advantage over general purpose computers (memory intensive)
  - e.g. PBKDF2 (Password-Based Key Derivation Function 2), scrypt, bcrypt, Argon2,...
- Disadvantage:
  - one cannot use the same hashed password file on a faster server and on an embedded device with an 8-bit microprocessor
    - need to use different values of x depending on the computational power of the machine
  - deemed too expensive for large Internet companies

Improving password security (2)

- Internet companies are using a function f "x" times with a small value of x combined with a MAC algorithm (e.g. HMAC).
  - idea: MAC computation with secret key in dedicated server
- Example Facebook (piling up of legacy systems)

SHA-2(bcrypt(HMAC$_K$(MD5(salt || password)))

Problem: human memory is limited



- Solution: store key K on magstripe, USB key, hard disk
- Stops guessing attacks

But this does not solve the other problems related to passwords

And now you identify the card, not the user….

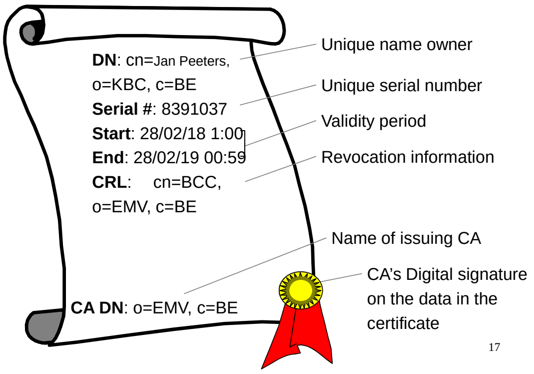
Possibility of replay: liveliness is missing

Improvement: Static Data Authentication

- Replace K by a signature of a third party CA (Certification Authority) on Alice's name: SigSK_{CA} (Alice) = special certificate
- Advantage: can be verified using a public string PK_{CA}
- Advantage: can only be generated by CA
- Disadvantage: signature = 40..128 bytes
- Disadvantage: can still be copied/intercepted

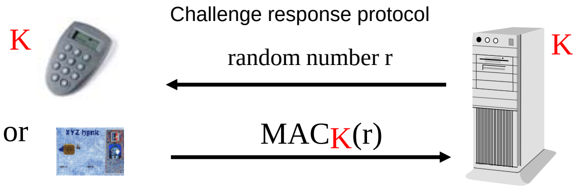
Possibility of replay: liveliness is missing

"Certificate" for static data authentication



DN: cn=Jan Peeters, o=KBC, c=BE
Serial #: 8391037
Start: 28/02/18 1:00
End: 28/02/19 00:59
CRL: cn=BCC, o=EMV, c=BE
CA DN: o=EMV, c=BE

Entity authentication with symmetric token



Challenge response protocol

K

or

random number r

MAC$_K$(r)

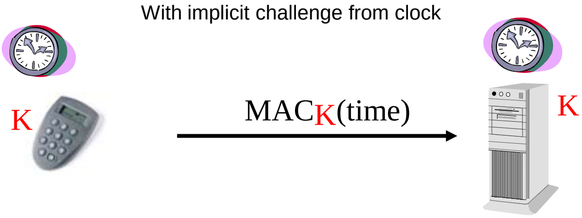
K

- Eavesdropping no longer effective
- Bob still needs secret key K

Detects whether Alice is alive!

Entity authentication with symmetric token

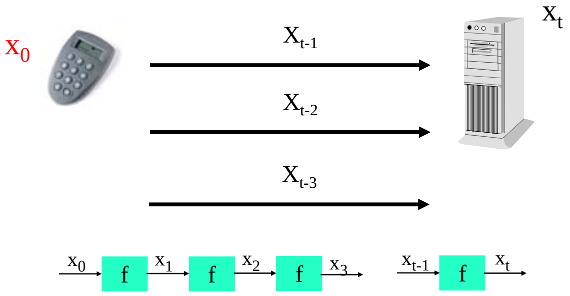
With implicit challenge from clock



- Eavesdropping no longer effective
- Bob still needs secret key K
- resynchronization mechanism needed

Lamport’s one-time passwords

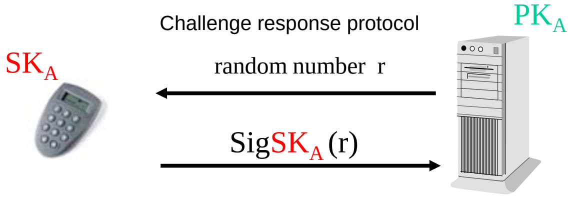
iterated one-way function



- Disadvantage: only works with one Bob

Entity authentication with public key token

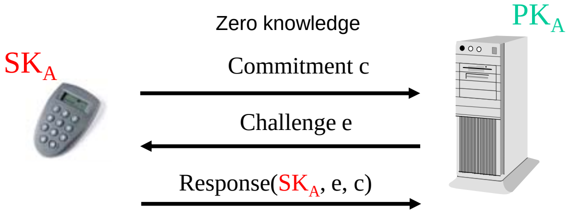
Challenge response protocol



- Eavesdropping no longer effective
- Bob no longer needs a secret – only PK_A

Entity authentication with ZK

Zero knowledge



- Mathematical proof that Bob only learns that he is talking to Alice (1 bit of information)
- Bob cannot use this information to convince a third party that he is/was talking to Alice

ZK definitions

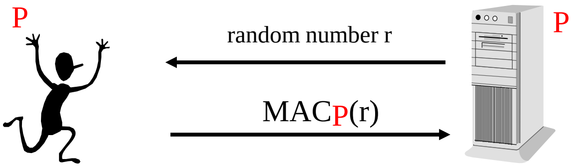
- **complete:** if Alice knows the secret, she can carry outthe protocol successfully
- **sound:** Eve (who wants to impersonate Alice) can only convinceBob with a very small probability that she is Alice;
- **zero knowledge:** even a dishonest Bob does not learn anything except for 1 bit (he is talking to Alice); he could have produced himself all the other information he obtains during the protocol.

Overview Identification Protocols

	Guess	Eavesdrop channel (liveliness)	Impersonation by Bob	Secret info for Bob	Mathema-tical proof	Security
Password	-	-	-	-	-	1
Magstripe (SK)	+	-	-	-	-	2
Magstripe (PK)	+	-	-	+	-	3
Dynamic password	+	+	-	-	-	4
Smart card (SK)	+	+	-	-	-	4
Smart Card (PK)	+	+	+	+	-	5
ZK	+	+	+	+	+	6

Entity authentication with password

Challenge response protocol

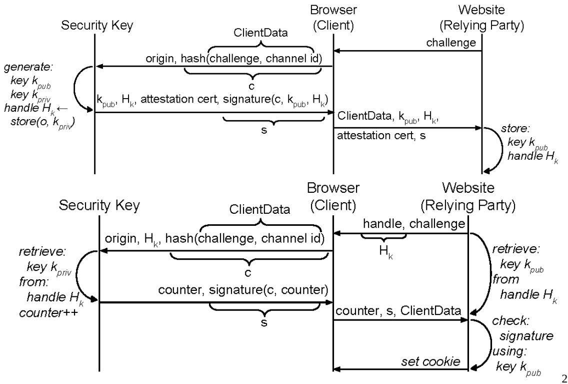


- Eavesdropping no longer effective
- Bob still needs secret key P
- Exhaustive search for P is easy based on a single transcript

Google's security keys

- Standardized by FIDO Alliance
- Threat model
  - web attackers (host malicious web content)
  - related site attackers
  - network level attackers
  - malware (but not in browser)
- Hardware: public key + button to press
- Generate key pair for each website and authenticate using device key pair

Google's security keys



Security Key

Browser (Client)

Website (Relying Party)

challenge

ClientData

origin, hash(challenge, channel id)

c

generate: *key k_{pub}* *key k_{priv}* *handle H_k ←* *store(o, k_{priv})*

k_{pub}, H_k, attestation cert, signature(c, k_{pub}, H_k)

s

ClientData, k_{pub}, H_k, attestation cert, s

store: *key k_{pub}* *handle H_k*

Security Key

Browser (Client)

Website (Relying Party)

handle, challenge

H_k

ClientData

origin, H_k, hash(challenge, channel id)

c

retrieve: *key k_{pub}* *from* *handle H_k*

retrieve: *key k_{priv}* *from:* *handle H_k* *counter++*

counter, signature(c, counter)

s

counter, s, ClientData

check: *signature* *using:* *key k_{pub}*

set cookie

Entity authentication in practice

- Phishing – mutual authentication
- Losing devices – local authentication to device – need to check proper linking of tw protocols (e.g. EMV)
- Sharing devices - biometry
- Interrupt after initial authentication – authenticated key establishment
- Mafia fraud – distance bounding

Mutual entity authentication

- Phishing is impersonating of the verifier (e.g. the bank)
- Most applications need entity authentication in two directions
- User needs to make judgment: difficult!
- Mutual entity authentication is not equivalent to 2 parallel unilateral protocols for entity authentication

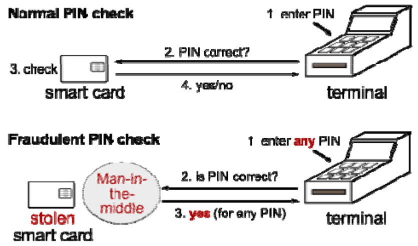
Limitations of devices

- Device authenticates user
  - but if the user looses the device…
  - solution: authenticate user to device using password, PIN or biometrics
  - but need to connect both phases properly! (EMV example)
- Device can be passed on to others (delegation, fraud)
  - solution: biometrics

Warning about EMV

http://www.cl.cam.ac.uk/research/security/banking/nopin/oakland10chipbroken.pdf

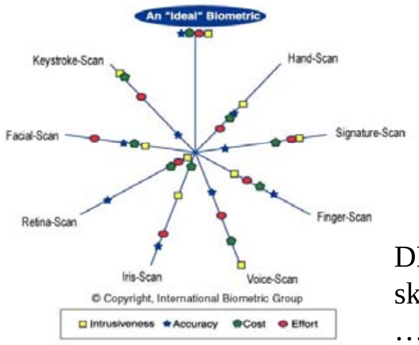
EMV PIN verification "wedge" vulnerability S.J. Murdoch, S. Drimer, R. Anderson, M. Bond, IEEE Security & Privacy 2010



Biometry

- Based on our unique features
- Identification or verification
  - Is this Alice?
  - Check against watchlist
  - Has this person ever registered in the system?

Some unique features



Source: Zephyr Analysis

DNA
skin
…

Biometric procedures

- Registration
- Template extraction

- Measurement
- Processing
- Template matching

- Link with applications

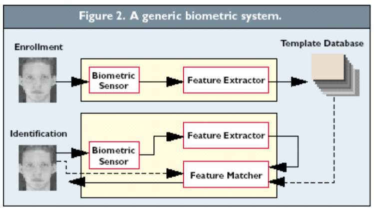
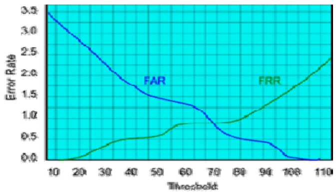


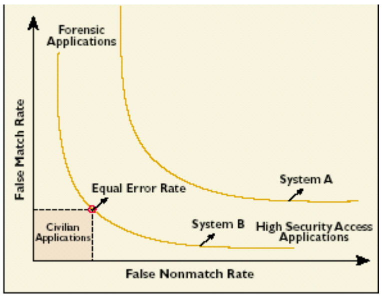
Figure 2. A generic biometric system.

Robustness/performance

- Performance evaluation
  - False Acceptance Ratio or False Match Rate
  - False Rejection Ratio or False Non-Match Rate
- Application dependent



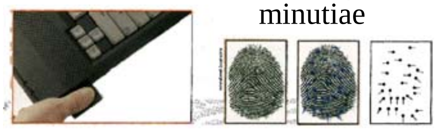
Robustness/performance (2)



Fingerprint

- Used for PC/laptop access
- Widely available
- Reliable and inexpensive
- Simple interface

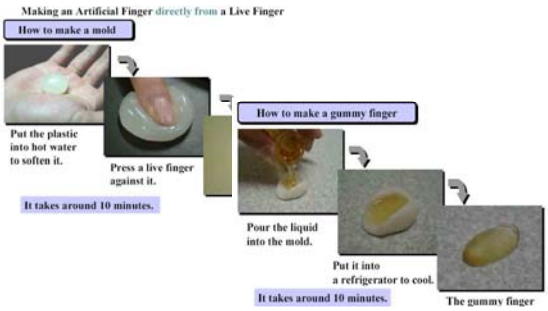
minutiae



Fingerprint (2)

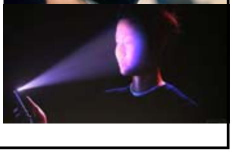
- Small sensor
- Small template (100 bytes)
- Commercially available
  - – Optical/thermical/capacitive
  - – Liveness detection
- Problems for some ethnic groups and some professions
- Connotation with crime

Fingerprint (3): gummy fingers



Facial recognition

- User friendly
- No cooperation needed
- Reliability improved in last few years
- Robustness issues
  - – Glasses/hair/beard/...
- E.g. Apple Face ID



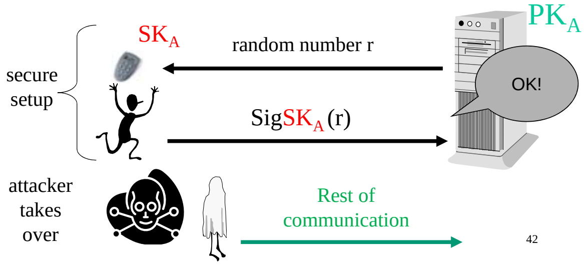
Biometry: pros and cons

- Real person
- User friendly
- Cannot be forwarded
- Little effort for user
- More suitable for supervised entity authentication (e.g. border controls)
- Evolving towards behavioral biometrics
- Secure implementation: derive key in a secure way from the biometric

- Privacy (medical)
- Intrusive?
- Liveliness?
- Cannot be replaced
- Risk for physical attacks
- Hygiene
- Does not work everyone, e.g., people with disabilities
- Reliability
- No cryptographic key

Keeping authenticity alive

- Establish who someone is
- Establish that this person is active/liveliness
- But what if the connection is broken after the initial phase?

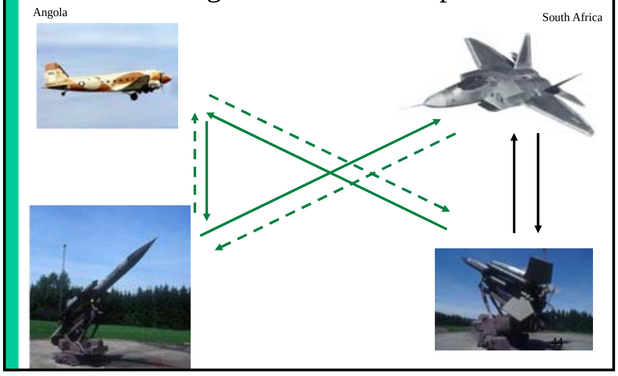


Solution

- Authenticated key agreement

- Run a mutual entity authentication protocol
- Establish a key
- Encrypt and authenticate all information exchanged using this key

The mafia fraud
– or the grandmaster chess problem



Location-based authentication

- Distance bounding: try to prove that you are physically close to the verifier

- Other uses of “location”
  - Dial-back: can be defeated using fake dial tone
  - IP addresses and MAC addresses can be spoofed
  - Mobile/wireless communications: operator knows access point, but how to convince others?
  - Trusted GPS: Galileo?

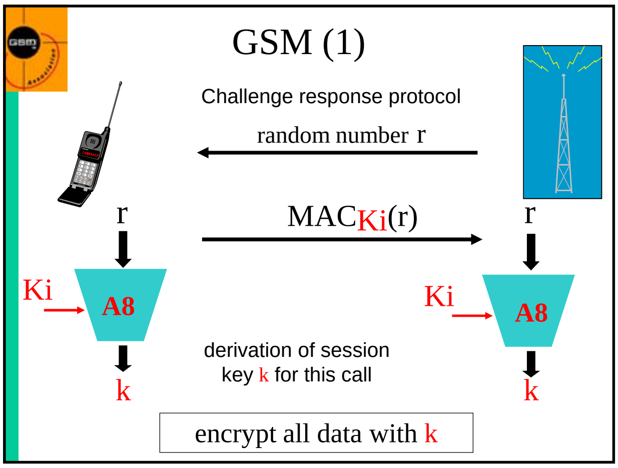
Key establishment

- The problem
- How to establish secret keys using secret keys?
- How to establish secret keys using public keys?
  - Diffie-Hellman and STS
- How to distribute public keys? (PKI)

Key establishment: the problem

- Cryptology makes it easier to secure information, by replacing the security of information by the security of keys
- The main problem is how to establish these keys
  - 95% of the difficulty
  - integrate with application
  - if possible transparent to end users

GSM (1)



Challenge response protocol

random number r

$\mathrm{MAC}_{Ki}(r)$

derivation of session key k for this call

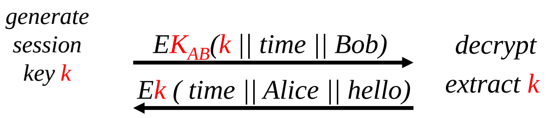
encrypt all data with k

GSM (2)

- SIM card with long term secret key Ki (128 bits)
- secret algorithms
  - A3: MAC algorithm
  - A8: key derivation algorithm
  - A5.1/A5.2: encryption algorithm
- anonimity: IMSI (International Mobile Subscriber Identity) replaced by TIMSI (temporary IMSI)
  - the next TIMSI is sent (encrypted) during the call set-up

Point-to point symmetric key distribution

Before: Alice and Bob share long term secret K_{AB}



generate session key k

$EK_{AB}(k \,||\, time \,||\, Bob) \rightarrow$

$\leftarrow Ek\,(time \,||\, Alice \,||\, hello)$

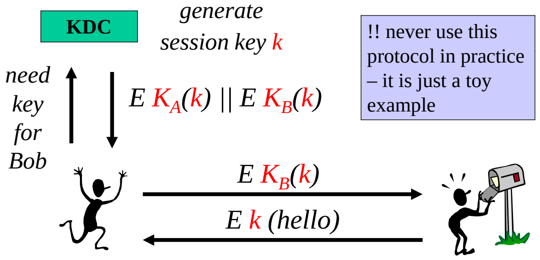
decrypt

extract k

- After: Alice and Bob share a short term key *k*
  - which they can use to protect a specific interaction
  - which can be thrown away at the end of the session
- Alice and Bob have also authenticated each other

Symmetric key distribution with 3rd party

Before (KDC=Key Distribution Center)
- Alice shares a long term secret with KDC: K_A
- Bob shares long term secret with KDC: K_B



KDC

generate session key k

need key for Bob

$E\ K_A(k)\ ||\ E\ K_B(k)$

!! never use this protocol in practice – it is just a toy example

$E\ K_B(k)$

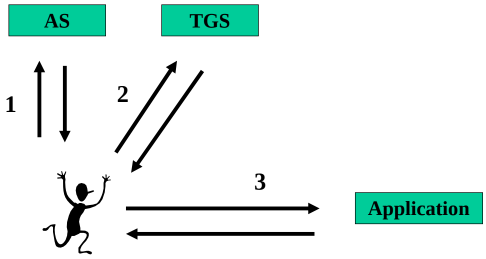
$E\ k\ (hello)$

Symmetric key distribution with 3rd party(2)

- After: Alice and Bob share a short term key ***k***
- Need to trust third party!
- Single point of failure in system

Kerberos/Single Sign On (SSO)

- Alice uses her password only once per day



Kerberos/Single Sign On (2)

- Step 1: Alice gets a "day key" K_A from AS (Authentication Server)
  - based on a Alice's password (long term secret)
  - K_A is stored on Alice's machine and deleted in the evening
- Step 2: Alice uses K_A to get application keys k_i from TGS (Ticket Granting Server)
- Step 3: Alice can talk securely to applications (printer, file server) using application keys k_i

A public-key distribution protocol: Diffie-Hellman

- Before: Alice and Bob have never met and share no secrets; they know a public system parameter α

generate x, *compute* α^x $\xrightarrow{\alpha^x}$ *generate* y, *compute* α^y

$\xleftarrow{\alpha^y}$

compute $k=(\alpha^y)^x$ *compute* $k=(\alpha^x)^y$

- After: Alice and Bob share a short term key k
  - Eve cannot compute $\mathbf{k}$: in several mathematical structures it is hard to derive x from α^x (this is known as the discrete logarithm problem)

Diffie-Hellman (continued)

generate x, *compute* α^x $\xrightarrow{\alpha^x}$ *generate* y, *compute* α^y

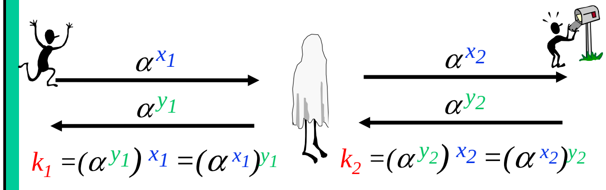
$\xleftarrow{\alpha^y}$

compute $k=(\alpha^y)^x$ *compute* $k=(\alpha^x)^y$

- BUT: How does Alice know that she shares this secret key $\mathbf{k}$ with Bob?
- Answer: Alice has no idea at all about who the other person is! The same holds for Bob.

Person-in-the middle attack

- Eve shares a key k_1 with Alice and a key k_2 with Bob
- Requires *active* attack



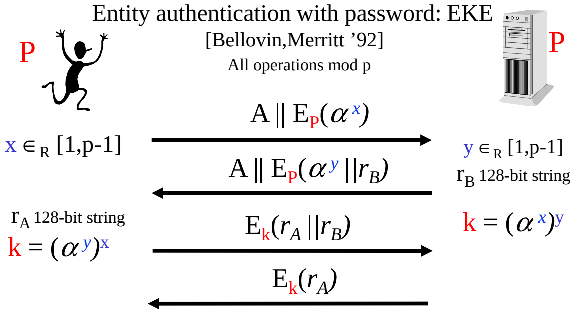
$\xrightarrow{\alpha^{x_1}}$ $\xleftarrow{\alpha^{y_1}}$ $\xrightarrow{\alpha^{x_2}}$ $\xleftarrow{\alpha^{y_2}}$

$k_1=(\alpha^{y_1})^{x_1}=(\alpha^{x_1})^{y_1}$ $k_2=(\alpha^{y_2})^{x_2}=(\alpha^{x_2})^{y_2}$

Entity authentication with password: EKE

[Bellovin,Merritt ’92]

All operations mod p



P P

$x \in_R [1,p-1]$ $\xrightarrow{A \,\|\, E_P(\alpha^x)}$ $y \in_R [1,p-1]$

$\xleftarrow{A \,\|\, E_P(\alpha^y \,\|\, r_B)}$ r_B 128-bit string

r_A 128-bit string $k=(\alpha^x)^y$

$k=(\alpha^y)^x$ $\xrightarrow{E_k(r_A \,\|\, r_B)}$

$\xleftarrow{E_k(r_A)}$

- Adds entity authentication to Diffie Hellman
- Attacker cannot perform off-line exhaustive search for the password P
- Attacker can still try on-line attacks; need to restrict number of uses of the account

- Literature: PAKE: Password Authenticated Key Establishment

Station to Station protocol (STS)

- The problem can be fixed by adding digital signatures
- This protocol plays a very important role on the Internet (under different names)

SK_A, PK_B SK_B, PK_A

choose x $\xrightarrow{\alpha^x}$ *choose* y

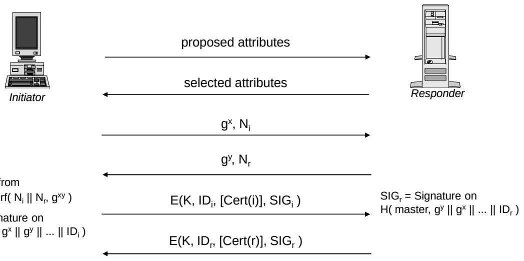
$\xleftarrow{\alpha^y}$

$k=(\alpha^y)^x$ $k=(\alpha^x)^y$

$\xrightarrow{SigA(\alpha^x \,\|\, \alpha^y)}$

$\sqrt{}\,SigB$ $\xleftarrow{SigB(\alpha^y \,\|\, \alpha^x)}$ $\sqrt{}\,SigA$

IKE - Main Mode with Digital Signatures



Initiator *Responder*

$\xrightarrow{\text{proposed attributes}}$

$\xleftarrow{\text{selected attributes}}$

$\xrightarrow{g^x,\ N_i}$

$\xleftarrow{g^y,\ N_r}$

K derived from
master = prf($N_i \,\|\, N_r$, g^{xy})

SIG_i = Signature on
H(master, $g^x \,\|\, g^y \,\|\, \ldots \,\|\, ID_i$)

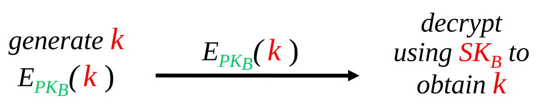
$\xrightarrow{E(K,\ ID_i,\ [Cert(i)],\ SIG_i\ )}$

SIG_r = Signature on
H(master, $g^y \,\|\, g^x \,\|\, \ldots \,\|\, ID_r$)

$\xleftarrow{E(K,\ ID_r,\ [Cert(r)],\ SIG_r\ )}$

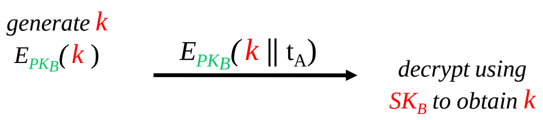
H is equal to prf or the hash function tied to the signature algorithm
(all inputs are concatenated)

Key transport using RSA


generate k
$E_{PKB}(k)$
$E_{PKB}(k)$
decrypt using SK_B *to obtain* k

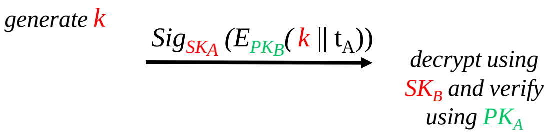
- How does Bob know that ***k*** is a fresh key?
- How does Bob know that this key ***k*** is coming from Alice?
- How does Alice know that Bob has received the key ***k*** and that Bob is present (entity authentication)?

Key transport using RSA (2)


generate k
$E_{PKB}(k)$
$E_{PKB}(k \| t_A)$
decrypt using SK_B *to obtain* k

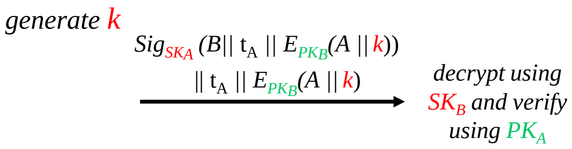
- Freshness is solved with a timestamp t_A

Key transport using RSA (3)


generate k
$Sig_{SKA}(E_{PKB}(k \| t_A))$
decrypt using SK_B *and verify using* PK_A

- Alice authenticates by signing the message
- There are still attacks (signature stripping…)

Key transport using RSA (4): X.509

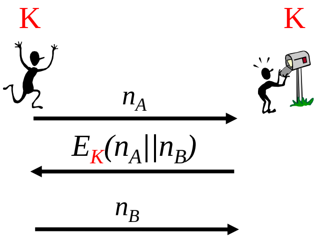

generate k
$Sig_{SKA}(B \| t_A \| E_{PKB}(A \| k))$
$\| t_A \| E_{PKB}(A \| k)$
decrypt using SK_B *and verify using* PK_A

Mutual: B can return a similar message including part of the first message

Problem (compared to D-H/STS): lack of **forward secrecy**

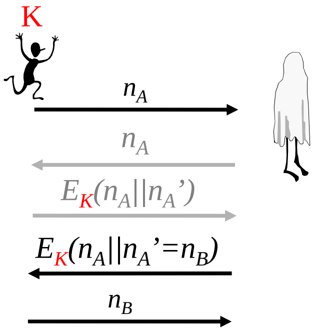
If the long term key SK_B of Bob leaks, all past session keys can be recovered!

A simple protocol



Reflection attack

Eve does not know K and wants to impersonate Bob



Conclusions

- Properties of protocols are subtle
- Many standardized protocols exist
  - ISO/IEC, IETF
- Difficulty: which properties are needed for a specific application

- Rule #1 of protocol design: **Don’t**
  - not even by simplifying existing protocols

Recommended reading: entity authentication

- NIST Special Publication 800-63 Version 1.0.2 (2006): Electronic Authentication Guideline: identifies four levels of assurance http://csrc.nist.gov/publications/nistpubs/800-63/SP800-63V1_0_2.pdf
- D. Balfanz, R. Chow, O. Eisen, M. Jakobsson, S. Kirsch, S. Matsumoto, J. Molina, P.C. van Oorschot: The Future of Authentication. IEEE Security & Privacy 10(1): 22-27 (2012)
- J. Bonneau, C. Herley, P.C. van Oorschot, F. Stajano: The Quest to Replace Passwords: A Framework for Comparative Evaluation of Web Authentication Schemes. IEEE Symposium on Security and Privacy 2012: 553-567
- J. Lang, A. Czeskis, D. Balfanz, M. Schilder, S. Srinivas, Security Keys: Practical Cryptographic Second Factors for the Modern Web. Financial Cryptography 2016: 422-440
- R. Peeters, J. Hermans, P. Maene, K. Grenman, K. Halunen, J. Häikiö, n-Auth: Mobile Authentication Done Right. ACSAC 2017: 1-15

See http://csrc.nist.gov/publications/PubsSPs.html
for about 120 Special Publications (800 Series) from NIST on computer security and cryptography

Recommended reading: key establishment

- A.J. Menezes, P.C. van Oorschot, S.A. Vanstone, Handbook of Applied Cryptography, CRC Press, 1997. Chapter 12.
- C. Boyd, A. Mathuria, Protocols for Authentication and Key Establishment. Information Security and Cryptography, Springer 2003, ISBN 978-3-642-07716-6.
- H. Krawczyk, SIGMA: The 'SIGn-and-MAc' Approach to Authenticated Diffie-Hellman and Its Use in the IKE-Protocols. CRYPTO 2003: 400-425.