Encryption Laws:
Evasion or Avoidance

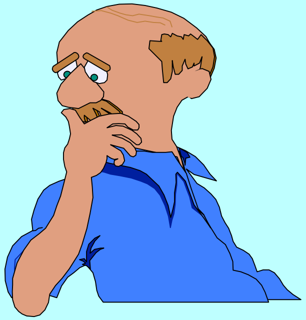
John A Bull jab@ansa.co.uk

Dave J Otway djo@ansa.co.uk

Structure

• Problem	Dave Otway
• Mechanisms	John Bull
• Solution	Dave Otway

The Dilemma



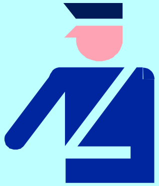
Strong encryption is regarded as essential for Electronic Commerce

There are legal constraints on the deployment of (strong) encryption

Constitutional Issues

- national security

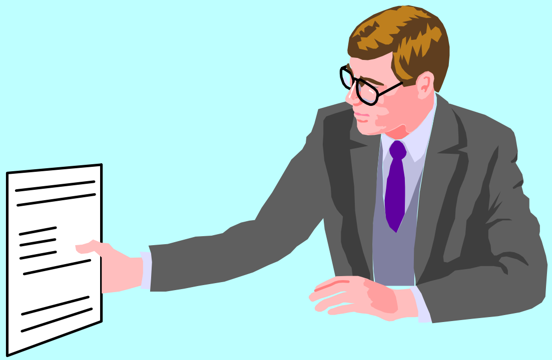
  ↳ export restrictions



- terrorism
  organised crime
  (one party) politics

  ↳ complete ban
  weakened use
  restricted use
  key escrow
  import restrictions

Commercial Issues



Patents

Copyright

Licensing

Globalisation Issues

- minimise number of (national) versions
  - ideally, only one each ➤ instead of 10^2
- minimise number of (international) pairings
  - ideally, only one ➤ instead of 10^4
- make mobile clients practical
  - no more than a handful ➤ potentially 10^6

The Usual Suspects

- agree on a standard solution
  - a political, not technical problem (UN/Gatt, not ISO)
- ignore the problem
  - carry on regardless, wait for somebody else to solve
- evade the authorities
  - lie, plead ignorance, chance prosecution, brazen it out
- avoid the problem
  - use another mechanism, re-exploit underlying maths
- minimise the problem
  - use encryption sparingly, pander to the main concerns

Preferred Solutions

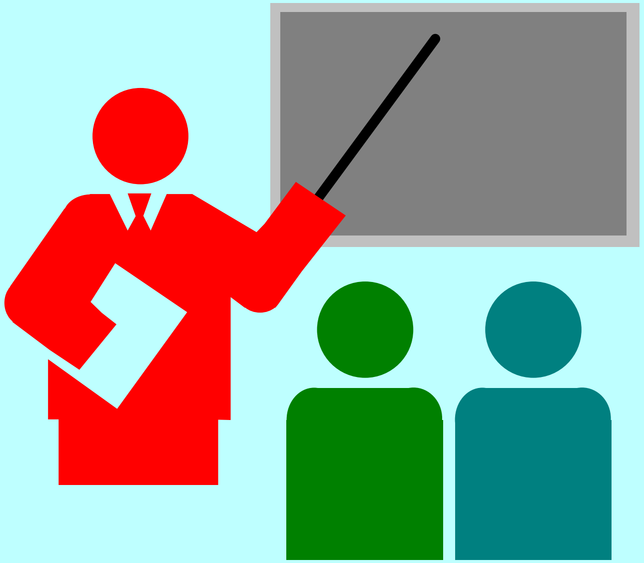
avoid encryption wherever possible

otherwise

minimise the amount of data encrypted

(ideally just random numbers - keys and checksums)

A Brief Tutorial
on Cryptographic Mechanisms



A Toolbox for a Solution

Boring crypto protocols

Soporific cryptobabble

Very hard mathematics

Back to school

Technology rules OK

The answer is 42

Standard stuff over again

One Way Functions

For $y = f(x)$

- Given x it is easy to compute y
- Given y it is very difficult to compute x

- Example:

$y = c^x$	$5^3 := ?$	$? = 125$
	$125 := 5^?$	$? = 3$

$x = \mathrm{Log}_c y$ $\qquad$ $\mathrm{Log}_5 125 := 3$

$x = \mathrm{Log}_{10} y / \mathrm{Log}_{10} c$

Finite, Integer Arithmetic

Multiplication modulo 7

0	0	0	0	0	0	0
0	1	2	3	4	5	6
0	2	4	6	1	3	5
0	3	6	2	5	1	4
0	4	1	5	2	6	3
0	5	3	1	6	4	2
0	6	5	4	3	2	1

One Way Functions in Cryptography

- Discrete logarithms
  - Diffie-Hellman $y = c^x \pmod{n}$
- Factorisation
  - RSA (mainly) $y=c.x \pmod{n}$
- Discrete polynomials
  - DSS (partially) $y = ax^n + bx^{n-1} + \dots + c \pmod{n}$

One Way Hash Functions

- Simple hash (shuffled data)

  Scrambled_block = Hash(block_of_data)

- Message digest (checksum)

  Fixed_sized_digest = Hash(block_of_data)

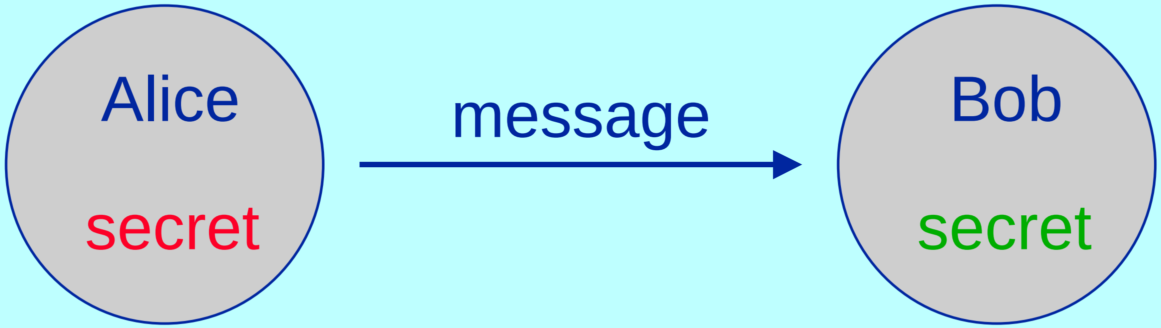
- Keyed digest (cryptographic checksum)

  Fixed_sized_digest = Hash(key, block_of_data)

Required Hash Function Properties

- H can be applied to a block of any size
- H produces a fixed length output
- H(x) is easy to compute given x
- Given v, it is infeasible to find x such that H(x)=v
- Given x, it is infeasible to find $y \neq x$ with H(y)=H(x)
- It is infeasible to find a pair (x, y) such that H(y)=H(x)

Hash Functions for Authentication



message = letter, H(secret, letter)

Does H(secret, letter) = H(secret, letter) ?

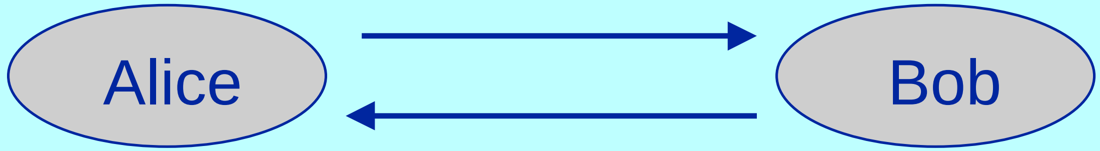
Authentication Protocol

- Is the sender who he claims to be?
  - Is the letter signed?
- Is the message that which he intended to send?
  - Is the letter sealed?
- Is the letter part of the present conversation?
  - Is the letter a “new” one?

Sent_message = nonce, letter, H(secret, letter, nonce)

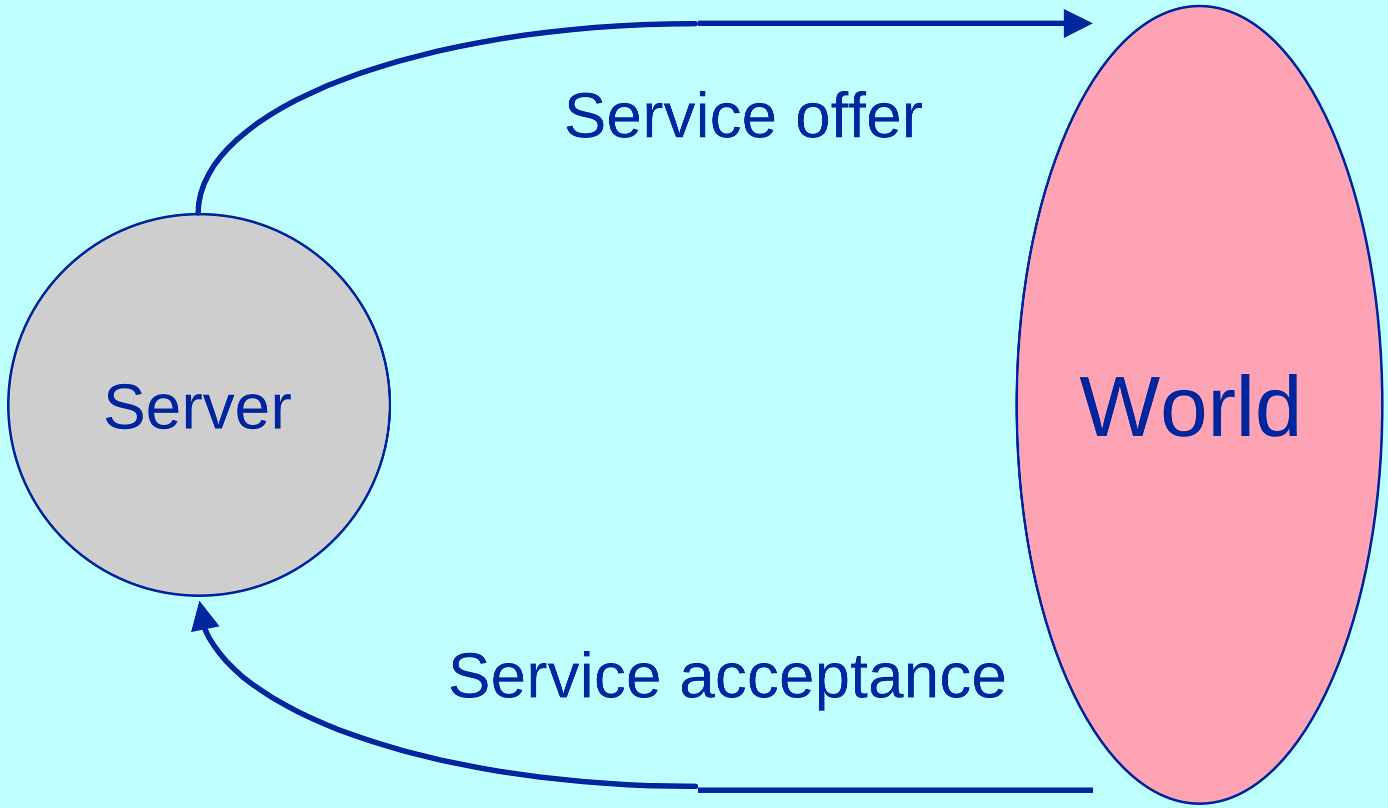
secret

nonce



Reply_message = nonce, reply, H(secret, reply, nonce)

Security in Practice

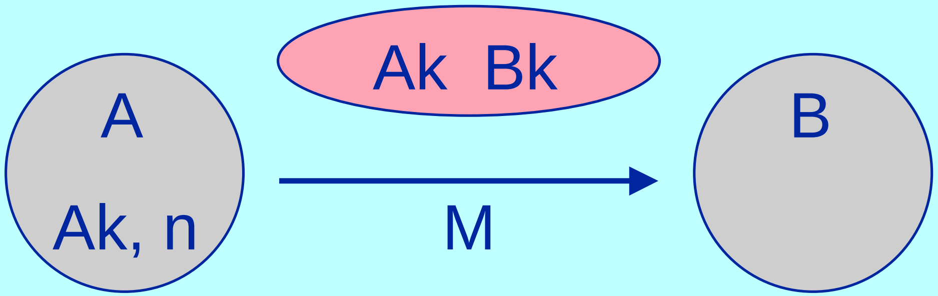


General Protocol

message = from_Alice, to_Bob, letter, nonce,
H(our_secret, to_Bob, letter, nonce)

but if a trusted third party (authentication server) holds the secrets (keys)

d = data
n = nonce
k = key



A = Alice
B = Bob

M = A, B, d, n, H(Ak, B, d, n)

Nested Protocol

A → B: [A, B, x, An, H(Ak, B, x, An)] = y

B → C: [B, C, y, Bn, H(Bk, C, y, Bn)] = z

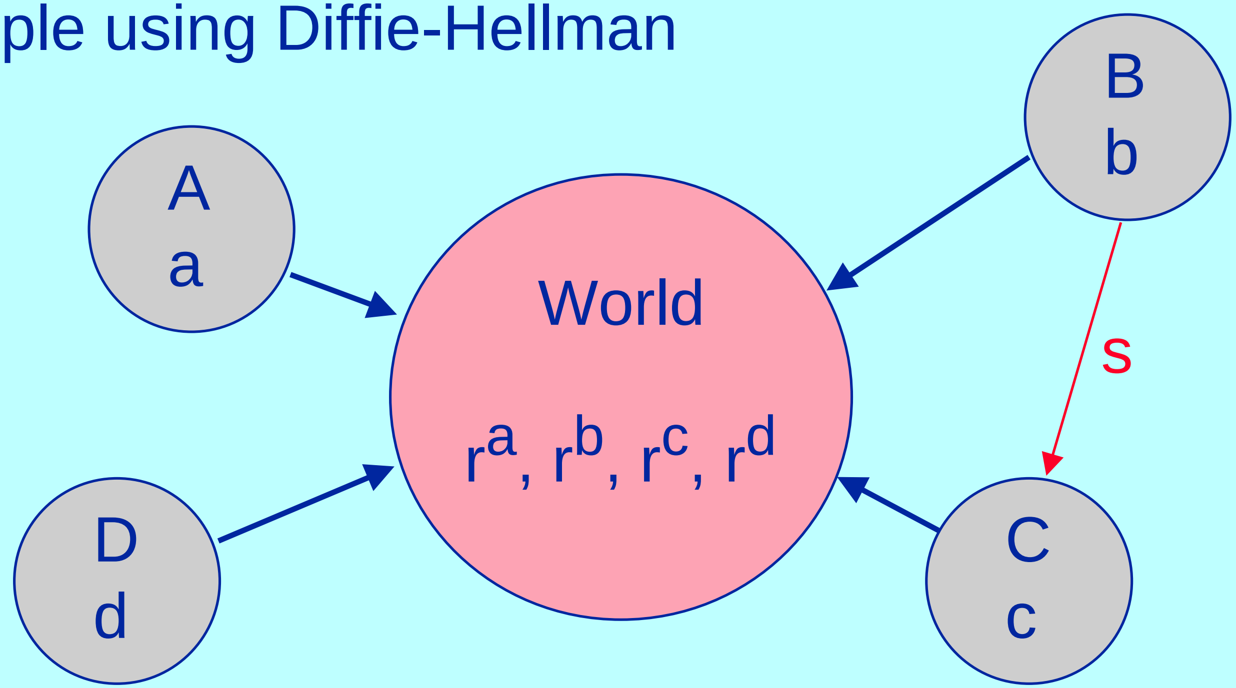
C → D: [C, D, z, Cn, H(Ck, D, z, Cn)] = etc

and include the use of a private secret

offer = service, H(service, private_secret)

Public Keys

Example using Diffie-Hellman



B → C: secret session key = s = $(r^b)^c = (r^c)^b = r^{bc}$

Public Key Protocol

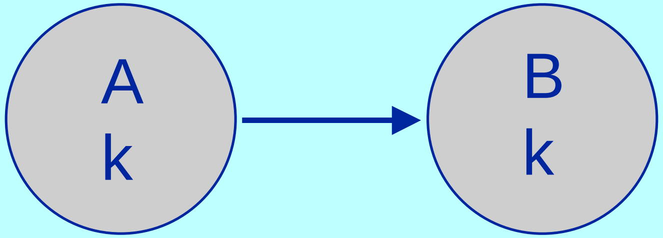
$A \rightarrow B$: $[A, B, x, An, H(r^{ab}, B, x, An)] = y$

$B \rightarrow C$: $[B, C, y, Bn, H(r^{bc}, C, y, Bn)] = z$

$C \rightarrow D$: $[C, D, z, Cn, H(r^{cd}, D, z, Cn)] = etc$

Key Distribution

Session_key = s
Master_key = k
Nonce = n
$\oplus$ = bitwise “exclusive or”



A $\rightarrow$ B: n, s $\oplus$ H(k, n), H(k, n, s)

s and n are generated at random;
n is sent “in clear”; s is “exclusive or’d” with H(k, n)
s is recovered from s $\oplus$ H(k, n)
s is checked using H(k, n, s)

Now Back to the Solution

Phew !!

Security Requirements

- key distribution ➤ how do we transmit keys
- integrity ➤ is this the message sent
- authentication ➤ who are we dealing with
- authorisation ➤ are they allowed to do this
- non-repudiation ➤ can they deny they sent this
- privacy ➤ do we care if anybody knows

Security Requirements

- key distribution ➤ how do we transmit keys
- integrity ➤ is this the message sent
- authentication ➤ who are we dealing with
- authorisation ➤ are they allowed to do this
- non-repudiation ➤ can they deny they sent this
- privacy ➤ do we care if anybody knows

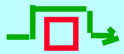
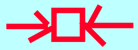
Key Distribution

- symmetric keys
  - master keys always physically distributed
  - secondary and session keys electronically distributed
    - new key XORed with digest of [nonce, master key]
    - Diffie-Hellman protocol
    - minimal encryption of [new key] with master key
- asymmetric keys
  - master public keys physically distributed or verified
  - secondary public keys electronically distributed
    - minimal encryption certificates verify new public keys

Security Requirements

- key distribution ➤ how do we transmit keys
- integrity ➤ is this the message sent
- authentication ➤ who are we dealing with
- authorisation ➤ are they allowed to do this
- non-repudiation ➤ can they deny they sent this
- privacy ➤ do we care if anybody knows

Integrity

- tamper proofing
  - seal with:
    - digest of [key, message, key] 
    - encrypted digest of [message] 
- replay prevention
  - include sequence number, or timestamp, in message
- loss detection
  - sequence number in message

Security Requirements

- key distribution ➤ how do we transmit keys
- integrity ➤ is this the message sent
- authentication ➤ who are we dealing with
- authorisation ➤ are they allowed to do this
- non-repudiation ➤ can they deny they sent this
- privacy ➤ do we care if anybody knows

Authentication

- proof of authorship
  by proving knowledge of a secret key
  - sign by:
    - digest of [key, message, key]
    - encrypted digest of [message]
- symmetric keys / asymmetric keys
  - symmetric keys require an on-line authentication service
  - asymmetric keys can be checked off-line with (encrypted) certificates

Security Requirements

• key distribution	➤	how do we transmit keys
• integrity	➤	is this the message sent
• authentication	➤	who are we dealing with
• authorisation	➤	are they allowed to do this
• non-repudiation	➤	can they deny they sent this
• privacy	➤	do we care if anybody knows

Authorisation

this requires no special security mechanisms

it is just a service that has to be secured

(by the same means as any other service)

a lack of privacy does not compromise its integrity

Security Requirements

• key distribution	➤ how do we transmit keys
• integrity	➤ is this the message sent
• authentication	➤ who are we dealing with
• authorisation	➤ are they allowed to do this
• non-repudiation	➤ can they deny they sent this
• privacy	➤ do we care if anybody knows

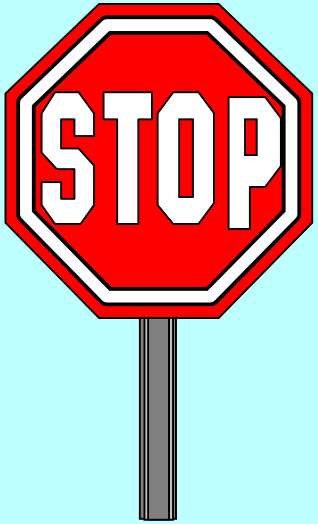
Non-repudiation

- replicated audit logs
  - legal agreements require audit logs to be kept by: (customer, issuer bank, merchant, acquirer bank, credit association, etc) so that fraud requires a conspiracy
- message certificates
  - on-line authentication service can verify:
    - digest of [key, message, key]
    - symmetrically encrypted digest of [message]
  - asymmetrically encrypted digest of [message] can be checked off-line with certificates

Security Requirements

- key distribution ➤ how do we transmit keys
- integrity ➤ is this the message sent
- authentication ➤ who are we dealing with
- authorisation ➤ are they allowed to do this
- non-repudiation ➤ can they deny they sent this
- privacy ➤ do we care if anybody knows

Privacy



OK

we give up

you can't have privacy without encryption

But ....

where you are banned from using encryption
(or you are only allowed to use weak encryption)

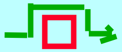
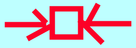
you can still have strong
key distribution, integrity, authentication
and non-repudiation

and you can deploy the same mechanisms everywhere

So ....

the security variations can be reduced to:

do we require off-line working ?

(avoid  or minimise  encryption)

what degree of privacy can be provided ?

The Bottom Line

the money is safe