CS4614 Monday 16th September 2013

About Network Protocols:
Talk securely accross a network.

Semester 1
- Cryptography (this is not a cryptography course)
  - How do we use it properly for secure applications.

Books

Matt Bishop - Intr to Com Sec
Dieter Gollmann - Com Sec
Jonathan Knudsen - Java Cryptography

Bruce Schneier - Applied Crytog.
Ross Anderson - Security Engineering
http://www.cl.cam.ac.uk/rja14/book.ht

http://security.stackexchange.com.
- Low level technical stuff in the form of Q&A.

week 7 and 12 - Tests.

CS4253 - Past Papers (Network & System 10 credit)

Semester 2 - CS4615 - System Security.

CS4614 Tuesday 17[th] September 2013

CS4615 Intro. ('SysSec) Semester 2 (Slides. No Notes)

CS4614 Moodle
CS4614 netsec (Key)

Principles.
- Any active entity that wants to generate or recieve a message
  - → Confidentiality
  - → Integrity (data can't be altered without detection)
  - → Availability
  - → Data Origin Authentication

- does not cover corruption over the network. If any issues are occur then the sender must be aware of the issue.

The above is what this module will cover.

Very Basic Cryptography.

$$D(K_2, E(K_1, P)) = P$$

E = Encryption
P = Plain Text
K_1 = Use K_1 to encrypt (KEY)
K_2 = Use K_2 to decrypt
D = Decrypt.

Semetric Cryptography ⇒ if ~~Encrypt~~ $K_1 = K_2$
Assemetric Cryptograpp ⇒ if $K_1 \neq K_2$

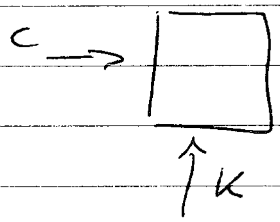
Substitution Cypher → sub with another
eg a – d, b – f so 'cat' = 'fdw'

Vegenére Cypher

message = attackatdawn
key = mykeymykeymy – repeats so a = y (twice)
cypher = cowsuic

One Time Pad = same lenght key as plain text.
so it will never repeat

Attack in depth = attacker guesses the key as they may know the sender always starts their message with mydear so they are able to guess the first six letters.



Key length = 52
∴ 2^{56} encryption
2^{256} 2^{512} etc

Three types of Encryption
DES – Data Encryption (1970's) ✗
AES – Advanced Encryption standard (2000's) ✓
RSA – Public Key Cipher (1970s)

✗ – can be broken – brute force – ✓ OK Cypher.

CS4614 Monday 23rd September 2013

$$D(K_2, E(K_1, P)) = P$$

$$E(K_1, P) = \text{Ciphertext}$$

Should look at having at least 120 bit encryption (eg DES) as a 56 bit for example could be broken in approx. 400 seconds using brute force whereas 128 bit would take forever with brute force tactics.

CloudCracker. (software online for cracking software)

Cut + Paste Attack

Bank PIN / Card – sign up and replace account name with yours on the attack card to trick the system into asking for your PIN instead of the actual account holders PIN.

Store [E (K3, (acctID, pin))] this will ensure that your card/pin is safe + secure.

Key stored on every ATM machine, not on a server for performance and network security issues.

CS4614 — Monday 30th September 2013

Symmetric Ciphers in Practice ("in the real world")

Encryption & Decryption ciphers use the same key.

Block Cipher – encrypt a block (chunk) at a time

eg.

DES	AES
64-bit block size (or 8 bytes)	128-bit 192-bit 256-bit

Standards are a good thing

- Kerckhoffs Principle (the attacker knows everything about the system except the key)
- Shannon's Maxim "the enemy knows the system"

plaintext → [Multiple transpositions & substitutions] → ciphertext block

↑ key

Double DES

0 → 55 56 → ///

$$C = E(K_1, E(K_2, P))$$

Encrypt plaintext with K_2 then encrypt that with K_1

Triple DES

$$C = E(K_1, D(K_2, E(K_1, P)))$$ ← 80 bit key.

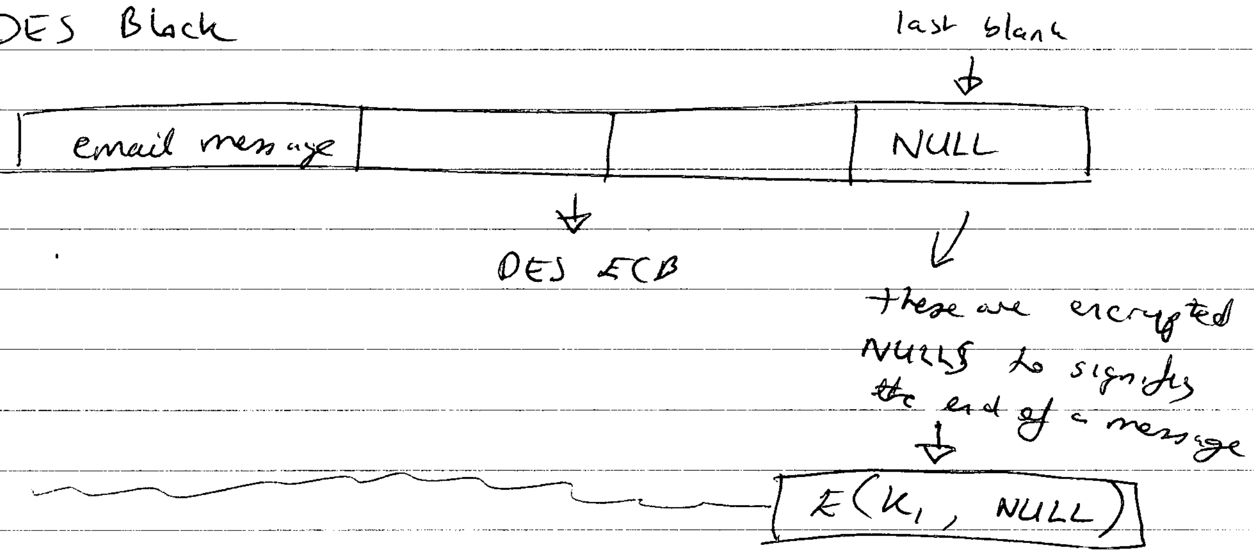
Encrypt Plaintext with K_1, then decrypt with K_2 and then encrypt with K_1.

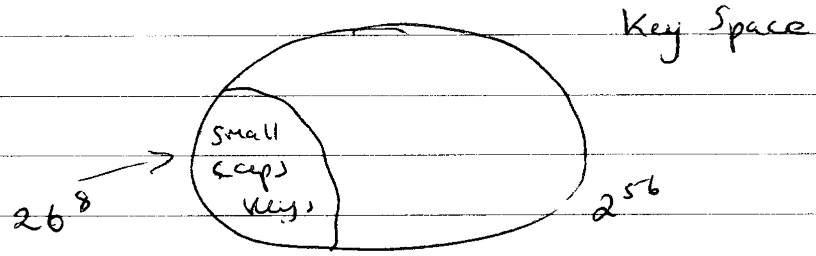
$\oplus$ XOR

Cut & Paste Attack

You have access to data, for example a students results file. You know from the position of the class list what position in the encrypted file your name is at. You also know the position of a student who you know got a great grade. You simply copy the encrypted grade and copy it to your grade. Thus you get a great grade.

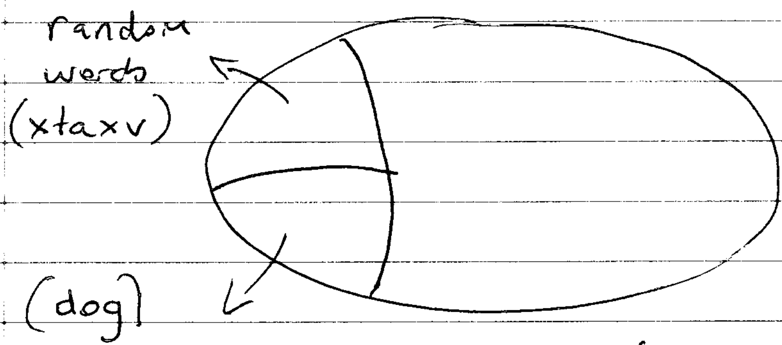
DES Block





There are only 26 lower case keys so try these first.

Another way is a Brute Force Dictionary Attack.

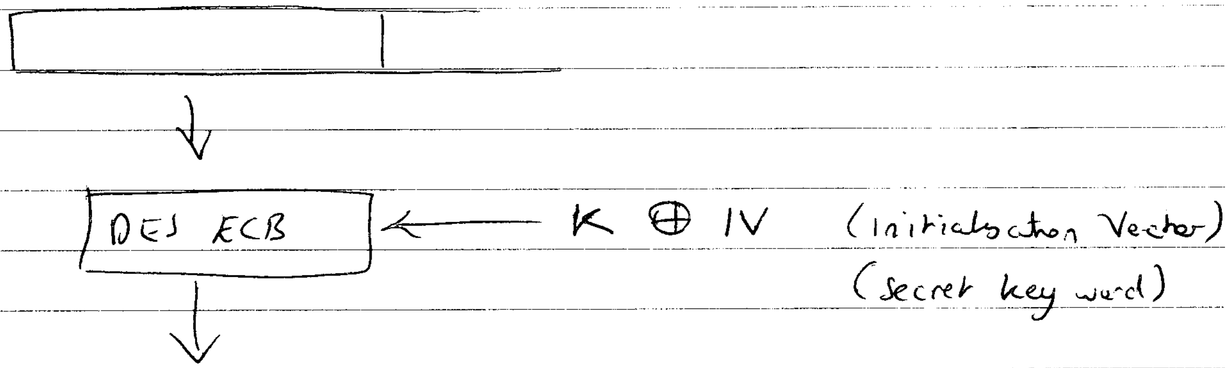


small section of 26^8 is an actual word in the dictionary

If the attacker can crack the nulls encryption they may have an insight into your encryption key.

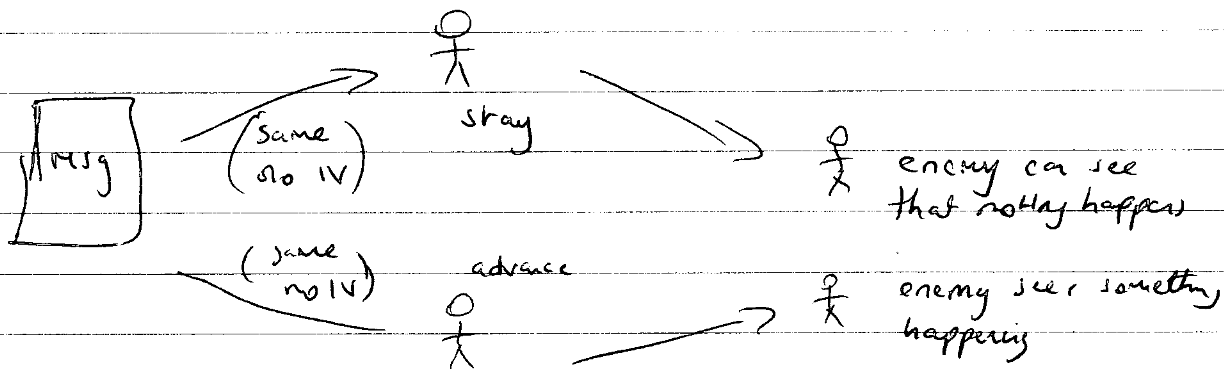
Add randomness to the encrypted message
- random word or value
- it will always look different.

Pizzas to the Pentagon Problem (1990 Gulf war)



Send message
that will look
different every time (use different IV each day)

you could also use IV_2 in the second block, IV_3 in the third but this is not very practical.



After a while when the enemy sees a certain encrypted message they can predict troop movements.

- OFB - Output Feedback Mode
- GCM - Galois / Counter Mode

CS4614 Tuesday 1st October 2013

4pm - 6pm - Monday Lab. - G21 / G19
Lab - 5 marks x2, 2 lab tests.

<u>Network Security</u>

OpenSSL - one of the best libraries

◇ MAC - Message Authentication Code.
MIC - Message Integrity Code

Swift 1 - mac key kaf
mac key kab

Banks primary concern is message integrity
Use MACs for integrity
MAC Keys managed end-to-end SWIFT
not trusted to manage MAC key.
- Swift - responsible for integrity, Banks
are responsible for encryption of message.

Swift 2 - uses public key crypto

Hash - always make <u>hard to reverse</u>.

MD5 is hard to reverse but it does not meet
the <u>collision freeness</u>.

SHA1 is much better (160 bits) 2^{63} } reverse
(2 x 160)
2^{63} - Collision

SHA 2 (256 and 512)

~~SEP~~.

★ $\{M\}_{K_{AB}}$ – encryption from now on
K

One way hash – Unpredictability !

CS4614 Monday 7th October 2013

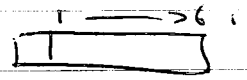
Java Cryptographic API's
Message Digest and Symmetric Ciphers

↓

One way Hash function

Message is always in Byte form (Byte Array).
MD5 = 28 bits
Byte Array = Binary Data.

base64 = Converts Binary Data to human readable code.

UTF8 ⇒ Extended ASCII (ASCII used to be 7 bits 
but it was extended to 8 bits to include
additional characters)

☆ md.reset() must be used if using it on a different file

HmacMD5
h_k(message) ≈ h(k ^ message)
↓ HmacMD5

PKCS5 Padding : if you have a message 4 bits long
what happens with the free space.
Padding out the free space is one
way.

SIMON = 6 bytes ⇒ 5 letters plus null (terminator)

CS4614 Monday 14th October 2014

Mid-Term in week 7.

Birthday Paradox

- Different Birthday

$1 - \frac{1}{365}$

Complexity

Passphrase Encryption PKCS #5

$\{M\}_K$

Take a ~~phrase~~ passphrase of any length and hash it. (One-way hash)

Additionally we will salt the passphrase and this becomes the key.

$$K = h^I(S \cdot P)$$

I should be greater than 1000

SALT ⇒ initialisation vector.

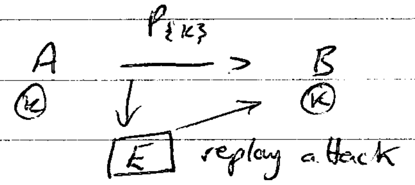
User Authentication

Important
One-Way Hash Functions
Block Ciphers

Authentication Mechanism
A user proving who they claim to be.
- Something that the user knows (eg PIN or secret)
- Something that the user has (eg Pass Card)
- Biometrics – fingerprint / iris

Often it could be a combination of these – MULTIFACTOR AUTHENITICATION – ATM Card & PIN

– One time password scheme – different password (value) every time – so it cannot be subjected to a replay attack.



Msg 1 : A → B : userid
Msg 2 : B → A : challenge
Msg 3 : A → B : $\{challenge\}_{K_{AB}}$

A can use password without revealing it.

Freshness: some value that has never been seen before, not random as random can repeat.

CS 4614 Monday 21st October 2013

Symmetric-key based Security Protocols

How does one system know the identity of a computer connecting to it.

eg Amazon → Credit Card details
(How are we sure we are giving them to amazon?)

- Public Key Crypto (also covered for rest of semester)

Entity Authentication

Msg 1 A → B : I'm Alice

Msg γ1 A[E] → B : I'm Alice ← Masquerade as Alice

Msg α1 A → B : I'm Alice
Msg β1 D → B : I'm Dan

We distinguish between different runs by using α, β.

A protocol is like a program, B (Bob) is the fileserver and Alice & Dan are trying to run it.

Msg 1   A → B : I'm Alice
Msg 2   B → A : R
Msg 3   A → B : $\{R\}_{K_{AB}}$

secret Key = K_{AB}
R is a nonce. number used only once. It provides a way for B to check the freshness of a Message i.e. whether he has seen it before.

This process is STATEFULL as B needs to keep track of challenges issued and responses between steps 2 and 3.

★ $\{\ldots\}_{K_{AB}}$ – denotes symmetric encryption using K_{AB} providing integrity and secrecy.

---

* α 1 A → B $\{$ I'm Alice, 7, 9.36 $\}_{K_{AB}}$
* β 1 A[E] → B : $\{$ I'm Alice, 7, 9.36 $\}_{K_{AB}}$

On receipt of α 1, Bob makes a note of nonce 7 for time window including 9.36 and since β 1 is within timewindow and nonce is same, he knows its a replay

α = alpha
β = beta

{ NONCE + TIME BASED RUN }

▲ Mutual Authentication : Alice authenticates Bob
Bob authenticates Alice

Alice knows she sent a message to Bob and Bob knows he got the message from Alice. and vice-versa. For example:

Msg 1 A -> B : I'm Alice, R_2
Msg 2 B -> A : R_1, $\{R_2\} K_{AB}$
Msg 3 A -> B : $\{R_1\} K_{AB}$

R_2 - Alices NONCE
R_1 - Bobs NONCE
K_{AB} - Key shared between Alice & Bob.

If [E] tries to replay this she will get as far as step 2 but cannot continue the run because she does not know the K_{AB}, but she manages to get a NONCE eg she gets $\{R_1\} K_{AB}$ so she is able to start a new run and use this information i.e. key and NONCE. Bob now thinks he is communicating with Alice, but instead he is communicat with Eve [E].

CS4614 Monday 5th November 2013

1/ A -> T : $\{B\}_{Kat}$, $\{K_{AB}\}_{Kat}$
2/ T -> B : $\{A\}_{Kbt}$, $\{K_{AB}\}_{Kbt}$

Does protocol protect K_{AB}?
Is it possible for Eve to discover K_{AB}?

It would be naive for A & T to assume that because Eve does not know the keys that are encrypting K_{AB} that it is secure.

Suppose Eve listens in on a previous run of protocol between A and B.

α1 A -> T : $\{B\}_{Kat}$, $\{K_{AB}\}_{Kat}$,
α2 T -> B : $\{A\}_{Kbt}$, $\{K_{AB}\}_{Kbt}$,

Eve is a participant and so can run protocol with Trent. Eve would like to trick Trent:

A[E] -> T : $\{E\}_{KAT}$, $\{K_{AB}\}_{Kat}$

Instruct Trent to forward key to Eve

Eve already has a copy of $\{K_{AB}\}_{Kat}$, and so she needs to get a copy of $\{E\}_{KAT}$

she has this

so just needs to get this

Eve requests to share a key with Alice (as Eve is able to do this) ⇒ β1 : E -> T : $\{A\}_{Ket}$, $\{K_{AB}\}_{Ket}$

$\beta 2$: $T \rightarrow A$: $\{E\}_{KAT}$, $\{KAB\}_{Kat}$

Trent just follows the protocol, Eve listens in and sees what she is looking for ie. $\{E\}_{KAT}$

Eve starts a new run of protocol

$\alpha 1$: $A [E] \rightarrow T$ $\{E\}_{KAT}$, $\{K_{AB}\}_{KAT}$

↑ Eve giving her name as Alice

→ Key to Alice (Eve).

$\alpha 2$: $T \rightarrow E$. $\{A\}_{Ket}$, $\{K_{AB}\}_{Ket}$

Now Eve (E) knows K_{AB}.

Eve hasn't broken any crypto (no attacks) she has just manipulated what she knows about the system to get the key (without actually knowing the key)

---

* (1) Is it possible for E to masquerade as A to B (even when A never initiates a key exchange with B)?

Reflection Attack! look this up.

CS4614 Monday 11th November 2013

Public Key Cryptography Overview

Public Key & Private Key.

- Alice signs message - anyone can confirm her signature by decrypting with her public key

Private Key K_A^{-1}
Public Key K_A

$C = \{P\}_K$

$P = \{C\}_{K^{-1}}$

P = Plaintext, C = Cipher-text, K = encryption key, K^{-1} decryption.

$A \rightarrow B \ \{\{\text{Message}\}_{K_B}\}_{K_B^{-1}}$

We have confidentiality, we now need authentication.

$A \rightarrow B \ \{\{\text{Message}\}_{K_A^{-1}}\}_{K_A}$

Combined

$A \rightarrow B : \{\{\text{message}\}_{K_A^{-1}}\}_{K_B}$

Anyone can read this so ← using her public key.

she encrypts it with Bobs public key and only Bob can decrypt it.

RSA Public Key Crypto (Sketch)

- Choose two large prime numbers p and q, let $n = p \times q$.
- Totient $\phi(n)$ - numbers less than n with no factors in common with n.
- Pick integer $e < n$ relatively prime to $\phi(n)$
  e = encryption, d = decryption
- Find a second integer d such that $e \times d \bmod n = 1$

△ It turns out that knowing $\phi(n)$ makes it easy / feasible to find this d.

Public key is (e, n), private key is (d, n)
let $m < n$

$c = m^e \bmod n$ - encryption - public key
$m = c^d \bmod n$ - decryption - private key.

example:
$p = 47$, $q = 71$, $n = pq = 3337$
$\phi(n) = (p-1)(q-1) = 3220$

$e = 79$, such that $\gcd(e, \phi n)) = 1$
$d = e^{-1} [\bmod \phi(n)] = \text{fastexp}(79, 3320 - 1, 3220)$
$= 1019$

To encrypt break into blocks
- 688232686 => 688 232 686

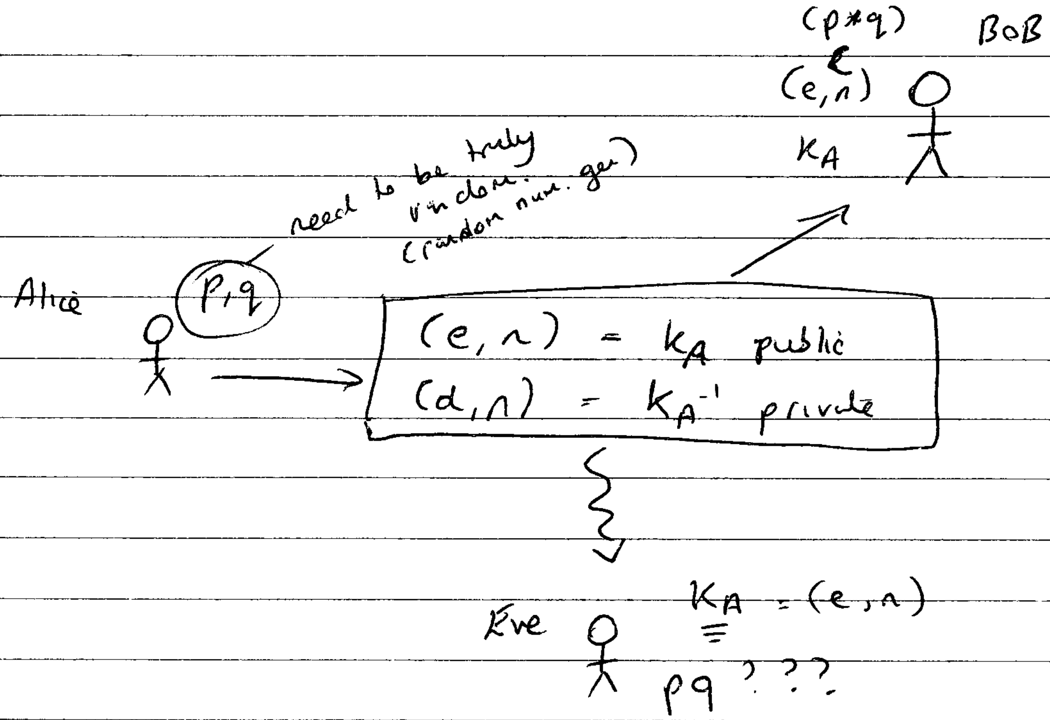
While feasible - it is quite costly.
We don't use RSA - we use symmetric key.

Once the primes have been assigned to pq, they must be destroyed as if an attacker know pq they can work out d. We also need to make sure pq are large enough.

private (d, n)
public (e, n)

In practice primes should be at least 1024 bit long. Use RSA implementation that are compliant with standards. Don't implement own RSA and don't use ones that are not compliant. (Website : RSA Laboratories)

* 768 bits was factored in approx. 3 years. From Jan '14 it should be 2048 bits (N.I.S.T.)



Example - OpenSSL weak pq generator, programmer error (commented out lines that was used to generate the pq) so entropy was poor.

CS4614 Monday 18th November 2013

Monday 6pm Lab - End of Term Test - 2nd December '13

Public Key Protocols : A Preliminary sketch of SSL.

Short term key : Session key
Could be for a single message of or for a few hours
An hour or Gb of data.

Server-side authentication - but server does not know who the person sending info is.

- What if attacker learns K_B^{-1}
  - The attacker keeps copies of all encrypted data $\{data\}_{KAB}$
  - If the attacker learns K_B^{-1}
  - Bob will come up with a new key
  - But the attacker will have access to all perevious data.
- This is called Perfect Forward Secrecy (PFS)
- DH does have " " " * DH = Diffie-Hellman

Station to Station Protocol (Diffe VanO. W)

Msg 2 B -> A ∴ Bob is saying we have encryption ~~by~~ but B is also saying he knows K (session key) so we also have authentication, (Secret key)

PFS + ~~Au~~Authentication + No timestamp

Some (Poor) Key Distribution Schemes.

Slide 3/33 -

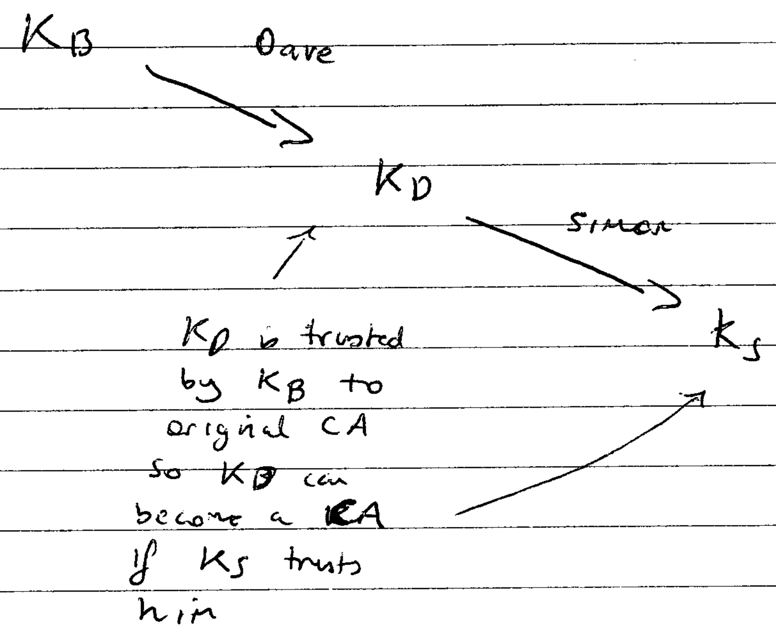
1. Bob can't be sure that it actually is Alice
2. The second way is also unusable

To ensure correct key is being used, we use Public Key Certificates. eg Trent

sK_T = signed by Trent

PKC's are third party (trusted) that confirm that the key is from the correct party.

Certification Authority (CA) guarantees to some degree.



CS4614 Monday 25th November 2013

Public Key Crypto.

Science of Info. hiding
Mathematical methods to encrypt.

private verify, public sign data.

DSA - Digital Signature Algorithm

Java Key store - holds keys and certificates

CSR : Certificate Signing Request.

Secure Socket Layer / Transport Layer Security
TLS is used pretty much by everyone these days
* Confidentiality & Authenticity

Symmetric Key -SSL - can be used both ways.

Msg 12 & 13 — change cipher text
⇒ basically telling each other to move into secure mode.

Always use factory management options when they exist.