Classical Encryption Algorithms:

Caesar/Affine Cipher:

> **Definition 1.4.4** Affine Cipher
> *Let* $x, y, a, b \in \mathbb{Z}_{26}$
> ***Encryption****:* $e_k(x) = y \equiv a \cdot x + b \bmod 26$.
> ***Decryption****:* $d_k(y) = x \equiv a^{-1} \cdot (y - b) \bmod 26$.
> *with the key:* $k = (a, b)$*, which has the restriction:* $\gcd(a, 26) = 1$.

The decryption is easily derived from the encryption function:

$$a \cdot x + b \equiv y \bmod 26$$
$$a \cdot x \equiv (y - b) \bmod 26$$
$$x \equiv a^{-1} \cdot (y - b) \bmod 26$$

The restriction $\gcd(a, 26) = 1$ stems from the fact that the key parameter a needs to be inverted for decryption. We recall from Sect. 1.4.2 that an element a and the modulus must be relatively prime for the inverse of a to exist. Thus, a must be in the set:

$$a \in \{1, 3, 5, 7, 9, 11, 15, 17, 19, 21, 23, 25\} \quad (1.2)$$

But how do we find a^{-1}? For now, we can simply compute it by trial and error: For a given a we simply try all possible values a^{-1} until we obtain:

$$a \cdot a^{-1} \equiv 1 \bmod 26$$

For an affine cipher mx + n (mod 26), we must have gcd(26,m) = 1, and we can always take 1 ≤ n ≤ 26.
– ϕ(26)= ϕ(2*13)=(2-1)*(13-1)= 12, hence we have 12*26=**312** possible keys.

Hill Cipher:

Αν ***P*** και ***C*** είναι διανύσματα διαστάσεως ***n*** που αναπαριστούν το plaintext και ciphertext, ενώ ***K*** είναι τετραγωνικός πίνακας ***n*** x ***n*** που αναπαριστά το κλειδί, τότε ο Hill Cipher μπορεί να αποδοθεί ως εξής:

- $C = E(K, P) = [P*K] \bmod 26$
- $P = D(K, C) = [C*K^{-1}] \bmod 26 = [P*K*K^{-1}] \bmod 26 = P$

Η δύναμη του έγκειται στο ότι κρύβει εντελώς τις συχνότητες εμφάνισης μονών γραμμάτων

- Η χρήση ενός μεγαλύτερου μεγέθους πίνακα κρύβει περισσότερες πληροφορίες συχνότητας.
- Αλγόριθμος κρυπτογράφησης Hill 3x3 κρύβει πληροφορίες συχνότητας όχι μόνο μεμονωμένων γραμμάτων, αλλά και ζευγών γραμμάτων.

Ισχυρός αλγόριθμος έναντι επιθέσεων ciphertext-only, αλλά μπορεί να σπάσει εύκολα με επιθέσεις known-plaintext.

Polyalphabetic Algorithms:

Βελτιώνουν την ασφάλεια με τη χρήση πολλαπλών αλφαβήτων κρυπτογράφησης.

Κάνουν την κρυπτανάλυση δυσκολότερη, καθώς περισσότερα αλφάβητα πρέπει να ανακτηθούν και αφού καμουφλάρεται η κατανομή συχνοτήτων.

Χρησιμοποιούν ένα κλειδί για να επιλέξουν ποιο αλφάβητο χρησιμοποιείται για κάθε γράμμα του μηνύματος.

Χρησιμοποιούν κάθε αλφάβητο με τη σειρά.

Επαναλαμβάνονται από την αρχή, μόλις φτάσει το τέλος του κλειδιού.

Συνοπτικά, αυτές οι τεχνικές έχουν τα ακόλουθα κοινά χαρακτηριστικά:

- Χρήση συνόλου σχετικών μονοαλφαβητικών κανόνων αντικατάστασης
- Ένα κλειδί καθορίζει ποιος συγκεκριμένος κανόνας επιλέγεται για ένα δεδομένο μετασχηματισμό

Vigenere:

Vigenère encryption E using the key K can be written,

$$C_i = E_K(M_i) = (M_i + K_i) \mod 26$$

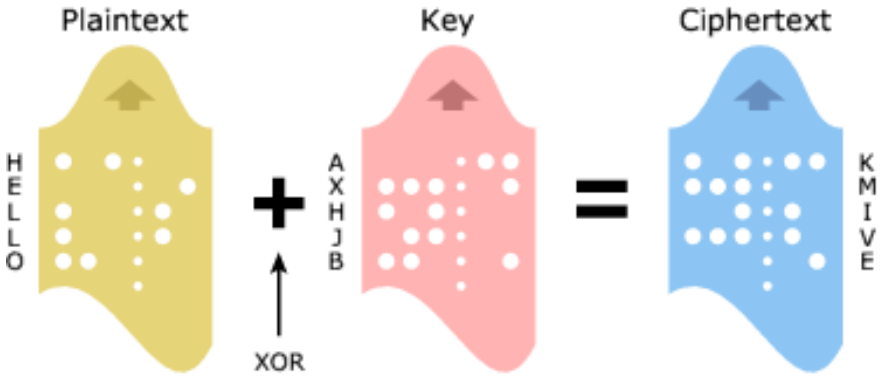
and decryption D using the key K,

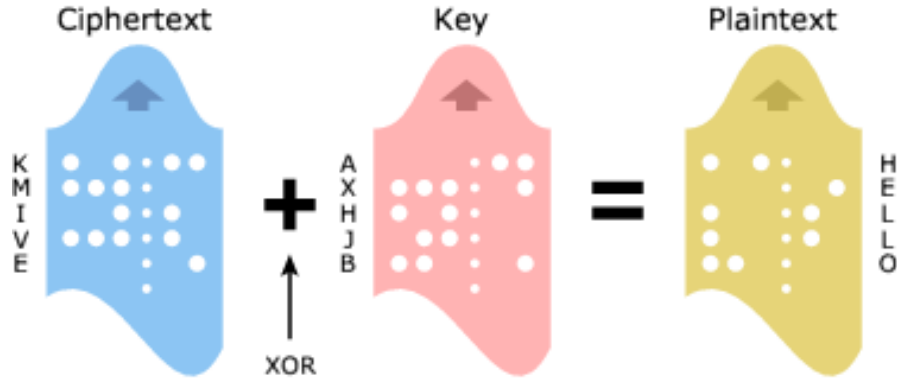
$$M_i = D_K(C_i) = (C_i - K_i) \mod 26,$$

Βασίζεται στο ότι εάν δύο όμοιες ακολουθίες γραμμάτων στο plaintext παρατηρηθούν σε απόσταση ίση με ακέραιο πολλαπλάσιο του μήκους του keyword length, θα παράγουν τις ίδιες ακολουθίες ciphertext.
Η κρυπτοανάλυση ξεκινά με την εύρεση όμοιων ακολουθιών γραμμάτων στο ciphertext (επαναλήψεις), τη συλλογή όλων των αποστάσεων μεταξύ τους και την αναζήτηση κοινών παραγόντων. Οπότε μπορεί να εντοπιστεί το ίδιο plaintext ένα ορισμένο διάστημα αργότερα, το οποίο έχει σαν αποτέλεσμα το ίδιο ciphertext, αποτελώντας ένδειξη για το μήκος του κλειδιού.

> CRYPTANALYSIS: KASISKI METHOD

Vernam:





Transposition/Permutation Algorithms:

Κρύβουν το μήνυμα αλλάζοντας τη σειρά των γραμμάτων.

Χωρίς να αλλάζουν τα γράμματα που χρησιμοποιούνται.

Μπορούν όμως να αναγνωριστούν τελικά, καθώς παρουσιάζουν την ίδια κατανομή συχνοτήτων εμφάνισης με το αρχικό κείμενο.

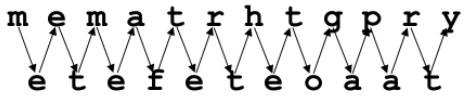
Rail Fence:

Τα γράμματα του μηνύματος γράφονται διαγώνια σε έναν ορισμένο αριθμό γραμμών.

Στη συνέχεια διαβάζεται το ciphertext κατά γραμμές.

Παράδειγμα:

- Το αρχικό μήνυμα "***meet me after the toga party***" γράφεται ως εξής (αν επιλέξουμε δύο γραμμές):



- οπότε προκύπτει το εξής ciphertext:

MEMATRHTGPRYETEFETEOAAT

Columnar Transposition Ciphers:

Γράφονται τα γράμματα του μηνύματος σε γραμμές, με έναν προκαθορισμένο αριθμό στηλών

Στη συνέχεια, διαβάζεται το ciphertext κατά στήλες, άλλα με διαφορετική σειρά στηλών, η όποια καθορίζεται από κάποιο κλειδί.

Παράδειγμα:

- key:
  4312567
- plaintext:
  attackpostponeduntiltwoam
- ciphertext:
  TTNAAPTMTSUOAODWCOIXKNLYPETZ

4	3	1	2	5	6	7
a	t	t	a	c	k	p
o	s	t	p	o	n	e
d	u	n	t	i	l	t
w	o	a	m	x	y	z

Product Ciphers:

Οι αλγόριθμοι αντικατάστασης ή αντιμετάθεσης δεν είναι ασφαλείς λόγω των χαρακτηριστικών της γλώσσας.

Για να αντιμετωπίσουμε το πρόβλημα αυτό χρησιμοποιούμε περισσότερους από έναν αλγορίθμους στη σειρά, αλλά:

- δύο αντικαταστάσεις κάνουν μια πιο σύνθετη αντικατάσταση
- δύο αντιμεταθέσεις κάνουν πιο σύνθετη αντιμετάθεση
- αλλά μια αντικατάσταση ακολουθούμενη από μια αντιμετάθεση (γνωστή ως ***Product Cipher***) δημιουργεί έναν νέο πολύ πιο δύσκολο αλγόριθμο κρυπτογράφησης

Αυτό αποτελεί τη γέφυρα μεταξύ κλασικών και σύγχρονων αλγορίθμων κρυπτογράφησης.

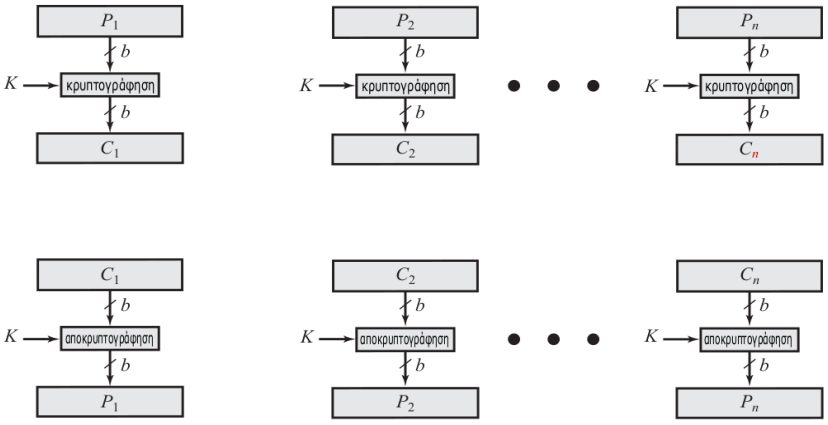
Εκ πρώτης όψεως η επαλήθευση ταυτότητας μπορεί να γίνει με συμμετρική κρυπτογράφηση:

- Θα μπορούσε ο Bob να πείσει την Alice για την προέλευση ενός μηνύματος M αν το στείλει ως $E[K, M]$ με την προϋπόθεση ότι
  - έχουν ανταλλάξει το κλειδί K μόνο μεταξύ τους και
  - εφόσον η Alice έχει ένα τρόπο να καταλάβει ότι το μήνυμα δεν έχει αλλοιωθεί.
- Η μη αλλοίωση μπορεί να εξασφαλισθεί περαιτέρω με ένα κώδικα ανίχνευσης λαθών, ενώ μπορεί να προστεθεί και αριθμός σειράς.
- Τέλος, μπορεί να προστεθεί και μια χρονοσφραγίδα.

Ωστόσο σκέτη συμμετρική κρυπτογράφηση κωδικοβιβλίου μπορεί να υποστεί επίθεση αναδιάταξης τμημάτων.

Electronic Codebook Method (ECB):

Μέθοδος του ηλεκτρονικού κωδικοβιβλίου (electronic codebook method):



Authentication:

Η επαλήθευση αυθεντικότητας μπορεί να γίνει με χρήση ενός *Κωδικού Αυθεντικοποίησης Μηνύματος* (Message Authentication Code, MAC).[2]

Προϋπόθεση είναι οι δύο πλευρές A, B να έχουν ανταλλάξει ένα κλειδί K_{AB} που χρησιμοποιείται μαζί με δεδομένο αλγόριθμο F.

Άρα η Alice στέλνει στον Bob το $X = (M, \text{MAC}_M)$, όπου

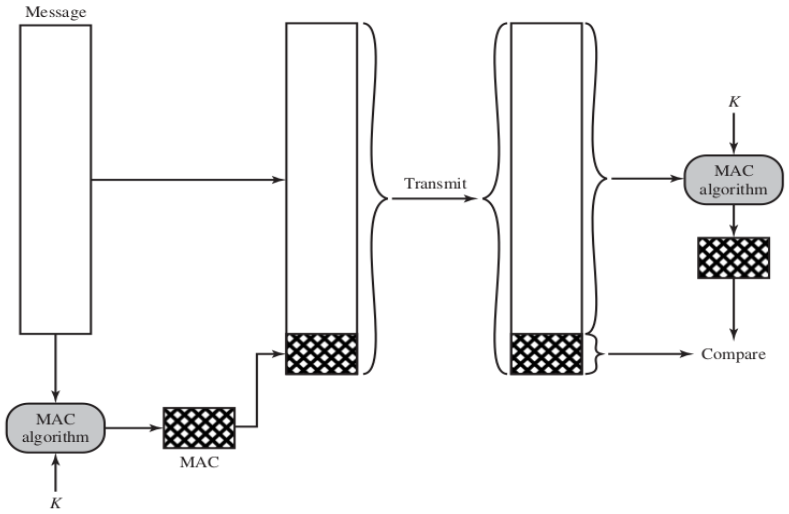
$$\text{MAC}_M = F(K_{AB}, M)$$

Ο Bob μέσα από ένα κανάλι επικοινωνίας λαμβάνει το Y, το οποίο μπορεί τουλάχιστον να διαχωρίσει ως $Y = (M', m)$.

Ο Bob ξαναϋπολογίζει το

$$\text{MAC}_{M'} = F(K_{AB}, M')$$

και αν $\text{MAC}_{M'} = m$ θεωρεί ότι $M' = M$ και ότι προέρχεται από την Alice.

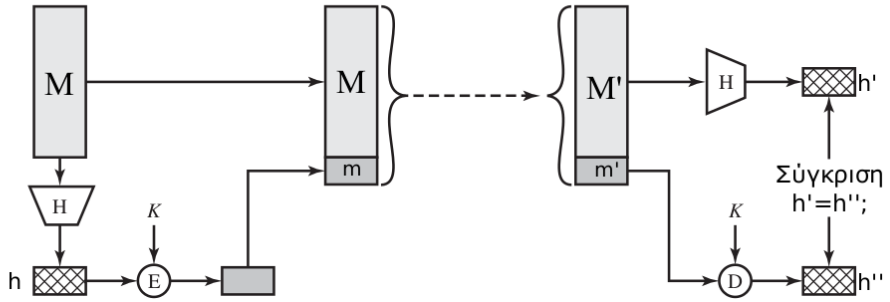


What we know:

1. Ότι ο μήνυμα δεν έχει αλλοιωθεί επειδή η Trudy δεν θα μπορούσε να δημιουργήσει το ίδιο MAC, παρ' όλο που μπορεί να γνωρίζει το M, εξ αιτίας του ότι δεν γνωρίζει το K_{AB}.
2. Ότι το μήνυμα προέρχεται από την Alice, που είναι η μόνη που γνωρίζει το K_{AB}.
3. Αν το μήνυμα περιλαμβάνει και αύξοντα αριθμό (όπως π.χ. στα X.25, HDLC, TCP), μπορεί να είναι βέβαιος ότι δεν έχασε ένα μήνυμα που ανήκει σε μια αριθμημένη σειρά μηνυμάτων.

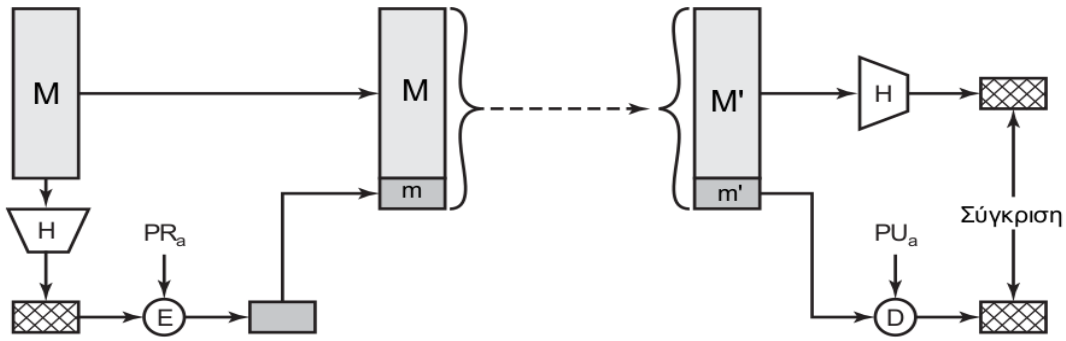
Hash function + Symmetric Encryption:

Χρήση συνάρτησης κατακερματισμού σε συνδυασμό με συμμετρική κρυπτογράφηση



- Ο Bob στέλνει το $M, E[K, H(M)]$.
- Η Alice συγκρίνει το $h' = H(M')$ με το $h'' = D[K, m']$.
- Για ποιους λόγους μπορεί να αποτύχει η σύγκριση;
- Τι καταλαβαίνει ο Bob από μια επιτυχημένη σύγκριση;

Hash function + Asymmetric Encryption (Public Key):



Public Key Encryption: (CONFIDENTIALITY)

Βασίζεται στα εξής βήματα:

1. Κάθε πλευρά παράγει ένα δημόσιο κι ένα αντίστοιχο ιδιωτικό κλειδί.
2. Κάθε πλευρά δημοσιοποιεί το δημόσιό της κλειδί.
3. Όταν ο Bob θέλει να στείλει μήνυμα στην Alice παίρνει το δημόσιο κλειδί της και κρυπτογραφεί το μήνυμα.
4. Στη συνέχεια η Alice αποκρυπτογραφεί το κρυπτογραφημένο μήνυμα με το ιδιωτικό της κλειδί.

Η διαδικασία αυτή έχει στόχο να εξασφαλίσει την *εμπιστευτικότητα*.

Authentication with Asymmetric Encryption:

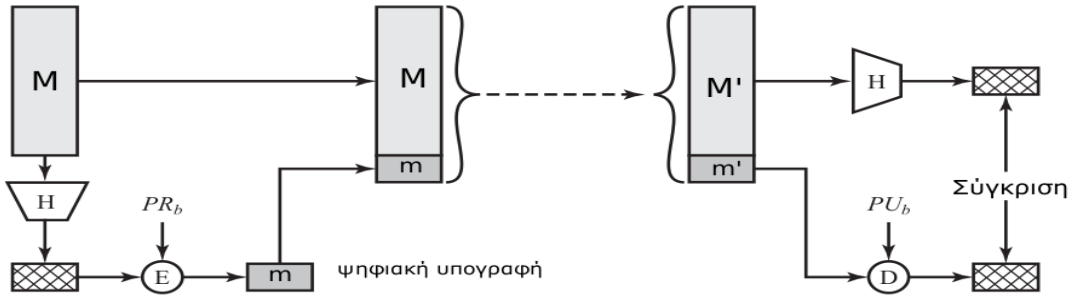
Βασίζεται στα εξής διαφορετικά από πριν βήματα:

3. Όταν ο Bob θέλει να στείλει μήνυμα στην Alice παίρνει το ιδιωτικό του κλειδί και το κρυπτογραφεί.
4. Στη συνέχεια η Alice αποκρυπτογραφεί το κρυπτογραφημένο μήνυμα με το δημόσιο κλειδί του Bob.

Παρατηρήσεις:

- Καθένας μπορεί να διαβάσει το μήνυμα του Bob, αλλά μόνο ο Bob μπορεί να το έχει στείλει.
- Η διαδικασία αυτή έχει στόχο να εξασφαλίσει την *αυθεντικότητα* της προέλευσης του μηνύματος.

Digital Signature:



- Με το σχήμα αυτό ο Bob στέλνει στην Alice ένα μήνυμα που σίγουρα προέρχεται απ' αυτόν.
- Η ψηφιακή υπογραφή δεν εξασφαλίζει εμπιστευτικότητα.

Block & Stream Ciphers:

Οι αλγόριθμοι τμημάτων (***Block Ciphers***) διαχωρίζουν το plaintext σε κομμάτια (ίσου μεγέθους), τα οποία επεξεργάζονται και το καθένα από τα οποία κρυπτογραφείται ή αποκρυπτογραφείται.

- Κρυπτογράφηση κάθε τμήματος σα μονάδα.
- Σαν αλγόριθμοι αντικατάστασης για πολύ μεγάλο πλήθος χαρακτήρων (=> ο καθένας απαιτεί 64 και πλέον bits για την αναπαράστασή του).

Οι αλγόριθμοι ροής (***Stream Ciphers***) όταν κρυπτογραφούν ή αποκρυπτογραφούν επεξεργάζονται ένα μόνο bit ή byte κάθε φορά.

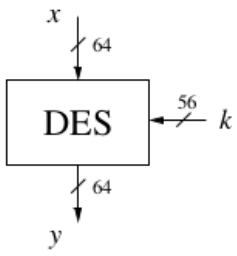
Πολλοί σύγχρονοι αλγόριθμοι κρυπτογράφησης είναι αλγόριθμοι τμημάτων.

- Αναλύονται καλυτέρα
- Έχουν ευρύτερο πεδίο εφαρμογών

Modes (ECB,CBC,CFB,OFB,CTR) of operation:

Mode	Description	Typical Application
Electronic Codebook (***ECB***)	Each block of 64 plaintext bits is encoded independently using the same key.	•Secure transmission of single values (e.g., an encryption key)
Cipher Block Chaining (***CBC***)	The input to the encryption algorithm is the XOR of the next 64 bits of plaintext and the preceding 64 bits of ciphertext.	•General-purpose block-oriented transmission
		•Authentication
Cipher Feedback (***CFB***)	Input is processed s bits at a time. Preceding ciphertext is used as input to the encryption algorithm to produce pseudorandom output, which is XORed with plaintext to produce next unit of ciphertext.	•General-purpose stream-oriented transmission
		•Authentication
Output Feedback (***OFB***)	Similar to CFB, except that the input to the encryption algorithm is the preceding DES output.	•Stream-oriented transmission over noisy channel (e.g., satellite communication)
Counter (***CTR***)	Each block of plaintext is XORed with an encrypted counter. The counter is incremented for each subsequent block.	•General-purpose block-oriented transmission
		•Useful for high-speed requirements

DES/3DES/DESX (Block Cipher):



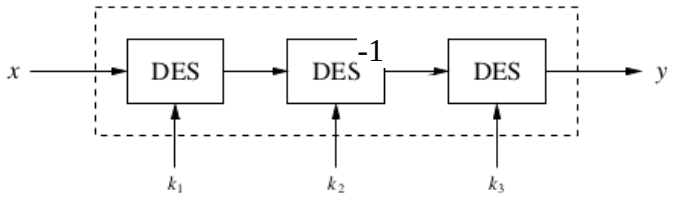
Input: 64 bits
Key: 56 bits
Output: 64 bits

Bruteforce (Exhaustive key search):

Definition 3.5.1 DES Exhaustive key search
Input*: at least one pair of plaintext–ciphertext* (x, y)
Output*: k, such that* $y = DES_k(x)$
Attack*: Test all* 2^{56} *possible keys until the following condition is fulfilled:*

$$DES_{k_i}^{-1}(y) \stackrel{?}{=} x \ , \ \ i = 0, 1, \ldots, 2^{56} - 1.$$

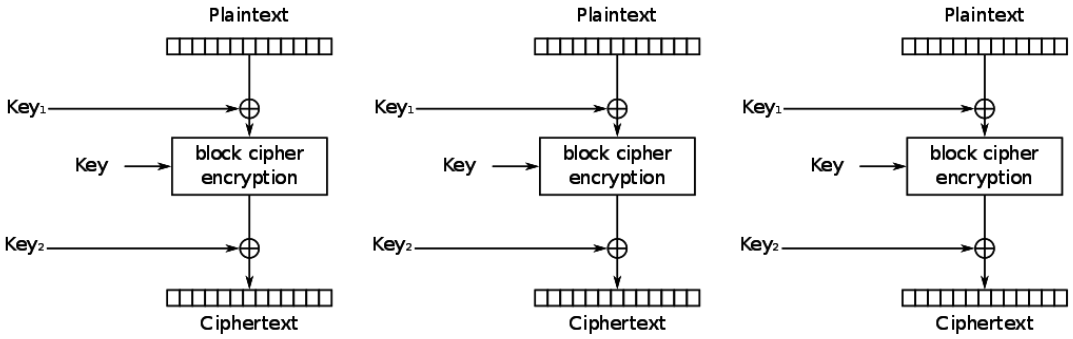
3DES (Backwards compatible with (simple) DES**):**



$$y = DES_{k_3}(DES_{k_2}^{-1}(DES_{k_1}(x)))$$

DESX (Key whitening**):**

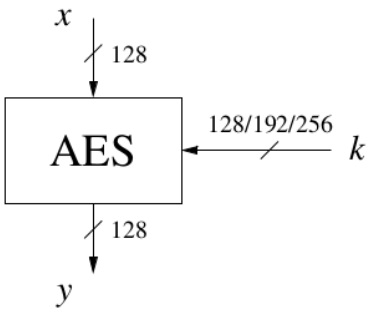
$$\text{DES-X}(M) = K_2 \oplus \text{DES}_K(M \oplus K_1)$$



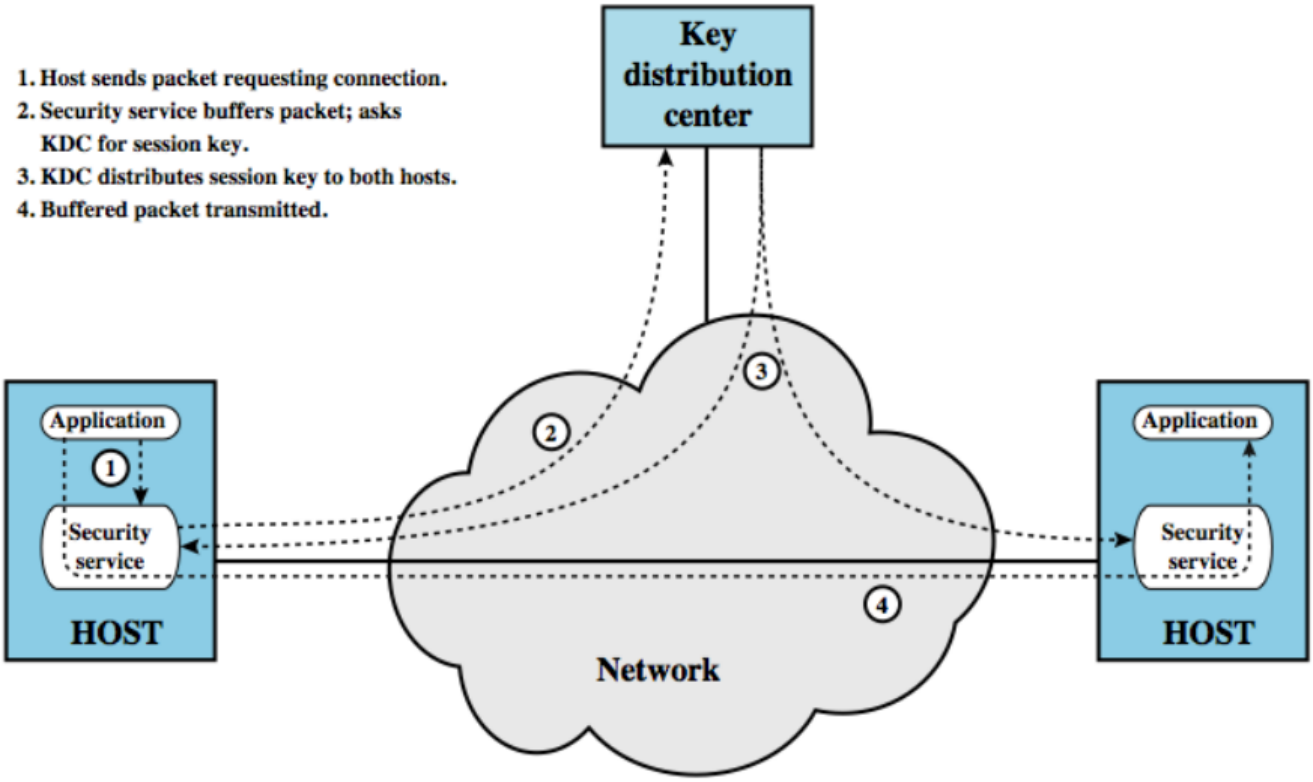
Xor Encrypt Xor (XEX) mode encryption

DES-X augments DES by XORing an extra 64 bits of key (K_1) to the plaintext *before* applying DES, and then XORing another 64 bits of key (K_2) *after* the encryption: The key size is thereby increased to 56 + (2 × 64) = 184 bits.

AES



Key Distribution (3rd party):



Number Theory (Basics):

Η εκτέλεση πρόσθεσης, αφαίρεσης και πολλαπλασιασμού μέσα στο σύνολο $\{0, 1, \ldots, n-1\}$ είναι δυνατή:

- $[(a \mod n) + (b \mod n)] \mod n = (a + b) \mod n$
- $[(a \mod n) - (b \mod n)] \mod n = (a - b) \mod n$
- $[(a \mod n) \times (b \mod n)] \mod n = (a \times b) \mod n$
- Ισχύει η ιδιότητα της απλοποίησης στην πρόσθεση:

$$(a + c) \equiv (b + c) \mod n \Rightarrow a \equiv b \mod n$$

- Όμως η απλοποίηση στον πολλαπλασιασμό είναι δυνατή μόνο με την προϋπόθεση οι c, n να είναι μεταξύ τους πρώτοι. Δηλαδή ισχύει

$$(a \times c) \equiv (b \times c) \mod n \Rightarrow a \equiv b \mod n$$

αν και μόνον αν $\text{μκδ}(c, n) = 1$.

RSA (Asymmetric Encryption):

- Ο RSA επεξεργάζεται τμήματα, όπου τόσο το αρχικό κείμενο M, όσο και το τελικό C είναι ακέραιοι μεταξύ 0 και $n - 1$.
- Η κρυπτογράφηση γίνεται μέσω του τύπου

$$C = M^e \mod n$$

- Η αποκρυπτογράφηση γίνεται ως

$$M = C^d \mod n$$

- Αμφότεροι αποστολέας και παραλήπτης γνωρίζουν τα n, e (e =δημόσιο κλειδί παραλήπτη).
- Μόνο ο παραλήπτης γνωρίζει το d (ιδιωτικό του κλειδί).

Η σχέση $M^{ed} \mod n = M$ ισχύει αν $ed \mod \phi(n) = 1$

Key Generation Algorithm (RSA):

- Διάλεξε δύο διαφορετικούς πρώτους αριθμούς $p.q$.
- Υπολόγισε το $n = pq$.
- Υπολόγισε το $\phi(n) = (p-1)(q-1)$.
- Διάλεξε ακέραιο e που να είναι πρώτος προς το $\phi(n)$.
- Υπολόγισε ένα d τέτοιο ώστε $de \bmod \phi(n) = 1$ (modular inverse of e).
- Το δημόσιο κλειδί είναι (e, n).
- Το ιδιωτικό κλειδί είναι (d, n).

RSA Encryption Example:

- Επιλέγουμε δύο πρώτους, $p = 19, q = 41$, άρα $n = pq = 779$ και $\phi(pq) = 18 \times 40 = 720$.
- Ζητάμε ακέραιους d, e ώστε $de \bmod 720 = 1$. Ας υποθέσουμε ότι κάνουν $de = 721$. Αναλύουμε σε πρώτους το $721 = 7 \times 103$, άρα μπορούμε να βάλουμε $d = 7, e = 103$ (ή αντίστροφα).
- Έστω ότι ο αποστολέας θέλει να στείλει $M = 80$ ($< n = 779$). Υπολογίζει το
$$C = 80^{103} \bmod 779 = 484$$
- Ο παραλήπτης υπολογίζει
$$M = 484^{7} \bmod 779 = 80, \quad OK$$

RSA Attack Example:

- Είναι δημοσιευμένα τα $n = 505891$ και $e = 307$. Μπορεί να βρεθεί το d;
- Ο επιτιθέμενος αναλύει το
$$n = 505891 = 521 \times 971$$
και υπολογίζει ότι $\phi = 520 \times 970 = 504400$.
- Αναλύει τα $k\phi + 1$. Το πρώτο απ> αυτά, το 504401, διαιρείται με 307 δίνοντας 1643, το οποίο μπορεί να δοκιμάσει αν κάνει για d.

RSA Attacks:

$n = pq$

$\Phi(n) = (p-1)(q-1) = pq - p - q + 1 = n - p - n/p + 1$

$p\Phi(n) = np - p^2 - n + p$

$p^2 - np + \Phi(n)p - p + n = 0$

$p^2 - (n - \Phi(n) + 1)\, p + n = 0$

Textbook RSA has no semantic security, therefore it is not secure against chosen plaintext attacks or ciphertext attacks. This is because, respectively, it is deterministic (encrypting the same message twice produces the same ciphertext) and multiplicatively homomorphic (an encrypted values can be multiplicatively modified under encryption).

Primitive Root of a Prime number:

- Δεδομένου ενός πρώτου αριθμού p, η αρχική του ρίζα a είναι ένας αριθμός τέτοιος ώστε οι δυνάμεις $a^1, a^2, \ldots, a^{p-1} \bmod p$ να είναι σε κάποια διάταξη ίσες με τους αριθμούς $1, 2, \ldots, p-1$.
- Ο επόμενος πίνακας δείχνει πώς συμβαίνει αυτό για $p = 13, a = 6$. Κάθε γραμμή περιέχει το $(k, a^k, a^k \bmod p)$.

1	6	6
2	36	10
3	216	8
4	1296	9
5	7776	2
6	46656	12
7	279936	7
8	1679616	3
9	10077696	5
10	60466176	4
11	362797056	11
12	2176782336	1

Diffie-Hellman Algorithm:

1. Δίνονται (δημόσια) ένας πρώτος αριθμός p και μια αρχική του ρίζα a.
2. Ο Bob επιλέγει ως ιδιωτικό κλειδί ακέραιο $X_B < p$ και υπολογίζει το δημόσιο κλειδί του ως $Y_B = a^{X_B} \bmod p$.
3. Η Alice επιλέγει ως ιδιωτικό κλειδί ακέραιο $X_A < p$ και υπολογίζει το δημόσιο κλειδί της ως $Y_A = a^{X_A} \bmod p$.
4. Ο Bob υπολογίζει το νέο κρυφό κλειδί $K_B = Y_A^{X_B} \bmod p$.
5. Η Alice υπολογίζει το νέο κρυφό κλειδί $K_A = Y_B^{X_A} \bmod p$.
6. Όμως $K_B = K_A = K$, οπότε το K μπορεί να χρησιμοποιηθεί ως γνωστό και στους δύο μυστικό κλειδί για συμμετρική κρυπτογράφηση.

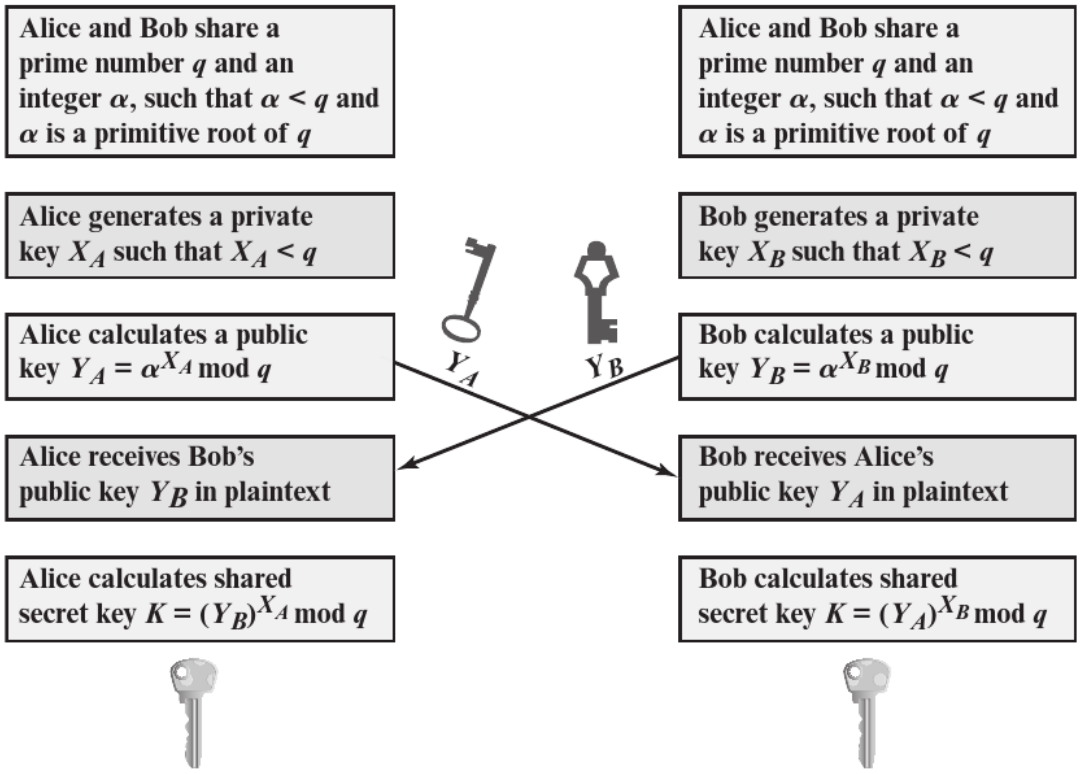
- Ισχύει η ιδιότητα: $(a \bmod b)^n \bmod b = a^n \bmod b$.
- Ισχύει επομένως ότι

$$\begin{aligned} K_A &= Y_B^{X_A} \bmod p = (a^{X_B} \bmod p)^{X_A} \bmod p \\ &= (a^{X_B})^{X_A} \bmod p \end{aligned}$$

- Ομοίως $K_B = (a^{X_A})^{X_B} \bmod p$, άρα $K_A = K_B$.
- Η Trudy διαθέτει τα p, a, Y_A, Y_B και πρέπει να υπολογίσει μια λύση της $Y_A = a^{X_A} \bmod p$ για να βρει το X_A και εξ αυτού το K (ή λύση της συμμετρικής της). Ο υπολογισμός αυτός είναι δύσκολος για μεγάλους ακέραιους.

Fermat's Little Theorem: $a^{p-1} \equiv 1 \pmod{p}.$

DISCRETE LOGARITHM CALCULATOR – https://www.alpertron.com.ar/DILOG.HTM
WOLFRAM ALPHA – https://www.wolframalpha.com/input/?i=



$$
\begin{aligned}
K &= (Y_B)^{X_A} \bmod q \\
&= (\alpha^{X_B} \bmod q)^{X_A} \bmod q \\
&= (\alpha^{X_B})^{X_A} \bmod q \\
&= \alpha^{X_B X_A} \bmod q \\
&= (\alpha^{X_A})^{X_B} \bmod q \\
&= (\alpha^{X_A} \bmod q)^{X_B} \bmod q \\
&= (Y_A)^{X_B} \bmod q
\end{aligned}
$$

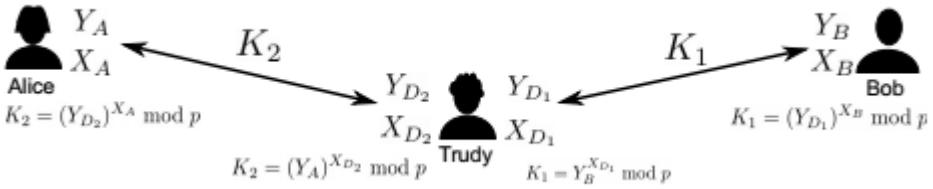
The result is that the two sides have exchanged a secret value. Typically, this secret value is used as shared symmetric secret key. Now consider an adversary who can observe the key exchange and wishes to determine the secret key K. Because X_A and X_B are private, an adversary only has the following ingredients to work with: q, α, Y_A, and Y_B. Thus, the adversary is forced to take a discrete logarithm to determine the key. For example, to determine the private key of user B, an adversary must compute

$$X_B = \text{dlog}_{\alpha,q}(Y_B)$$

The adversary can then calculate the key K in the same manner as user B calculates it. That is, the adversary can calculate K as

$$K = (Y_A)^{X_B} \bmod q$$

MitM (Man in the middle attack):



Το πρωτόκολλο είναι ευάλωτο σ> αυτή την επίθεση διότι δεν κάνει έλεγχο ταυτότητας των δύο πλευρών. Η αδυναμία αυτή μπορεί να διορθωθεί με τη χρήση ψηφιακών υπογραφών και πιστοποιητικών δημόσιου κλειδιού.

Key Distribution:

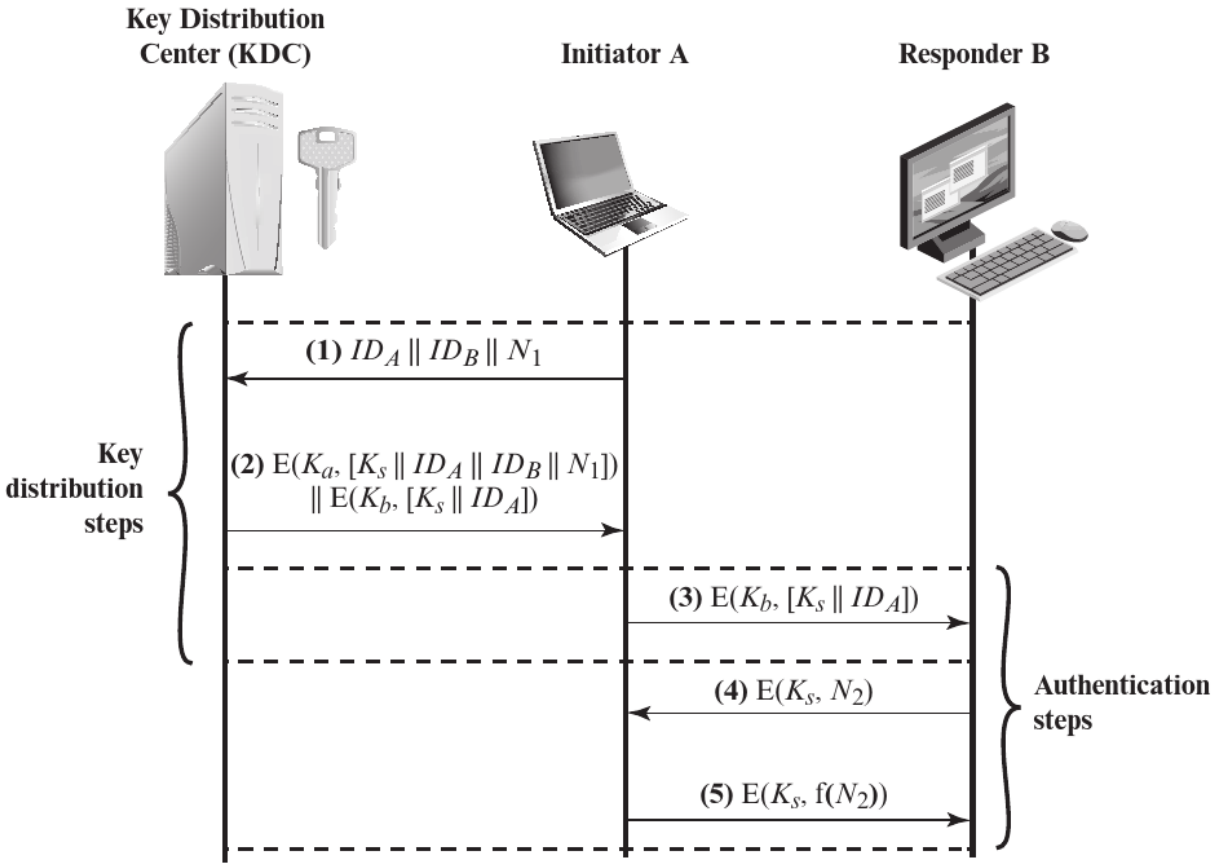


Figure 14.3 Key Distribution Scenario

A has a master key, K_a, known only to itself and the KDC; similarly, B shares the master key K_b with the KDC. The following steps occur.

1. A issues a request to the KDC for a session key to protect a logical connection to B. The message includes the identity of A and B and a unique identifier, N_1, for this transaction, which we refer to as a **nonce**. The nonce may be a timestamp, a counter, or a random number; the minimum requirement is that it differs with each request. Also, to prevent masquerade, it should be difficult for an opponent to guess the nonce. Thus, a random number is a good choice for a nonce.
2. The KDC responds with a message encrypted using K_a. Thus, A is the only one who can successfully read the message, and A knows that it originated at the KDC. The message includes two items intended for A:
   - The one-time session key, K_s, to be used for the session
   - The original request message, including the nonce, to enable A to match this response with the appropriate request

   Thus, A can verify that its original request was not altered before reception by the KDC and, because of the nonce, that this is not a replay of some previous request.

   In addition, the message includes two items intended for B:
   - The one-time session key, K_s, to be used for the session
   - An identifier of A (e.g., its network address), ID_A

   These last two items are encrypted with K_b (the master key that the KDC shares with B). They are to be sent to B to establish the connection and prove A's identity.

3. A stores the session key for use in the upcoming session and forwards to B the information that originated at the KDC for B, namely, $E(K_b,[K_s \| ID_A])$. Because this information is encrypted with K_b, it is protected from eavesdropping. B now knows the session key (K_s), knows that the other party is A (from ID_A), and knows that the information originated at the KDC (because it is encrypted using K_b).

At this point, a session key has been securely delivered to A and B, and they may begin their protected exchange. However, two additional steps are desirable:

4. Using the newly minted session key for encryption, B sends a nonce, N_2, to A.
5. Also, using K_s, A responds with f(N_2), where f is a function that performs some transformation on N_2 (e.g., adding one).

These steps assure B that the original message it received (step 3) was not a replay.

Note that the actual key distribution involves only steps 1 through 3, but that steps 4 and 5, as well as step 3, perform an authentication function.

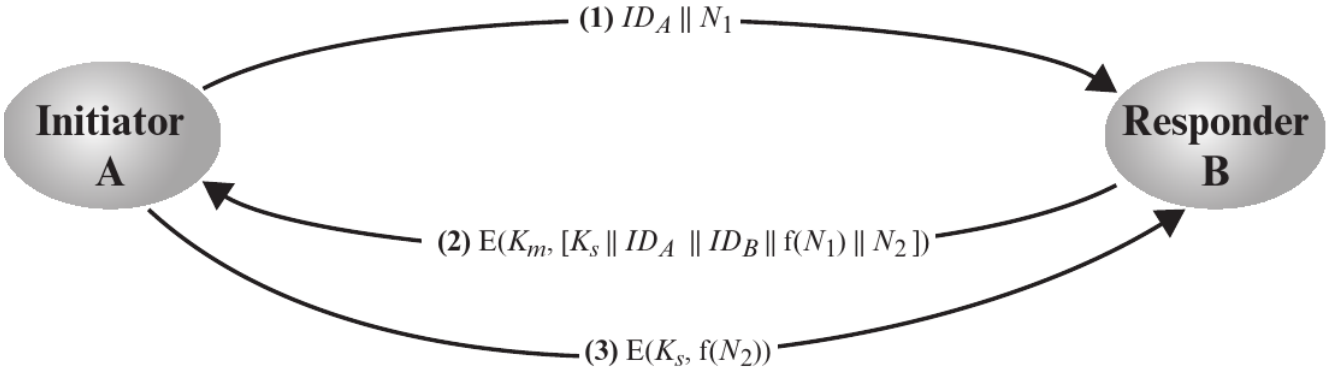
Decentrilized Key Distribution:

The use of a key distribution center imposes the requirement that the KDC be trusted and be protected from subversion. This requirement can be avoided if key distribution is fully decentralized. Although full decentralization is not practical for larger networks using symmetric encryption only, it may be useful within a local context.

A decentralized approach requires that each end system be able to communicate in a secure manner with all potential partner end systems for purposes of session key distribution. Thus, there may need to be as many as $[n(n-1)]/2$ master keys for a configuration with n end systems.

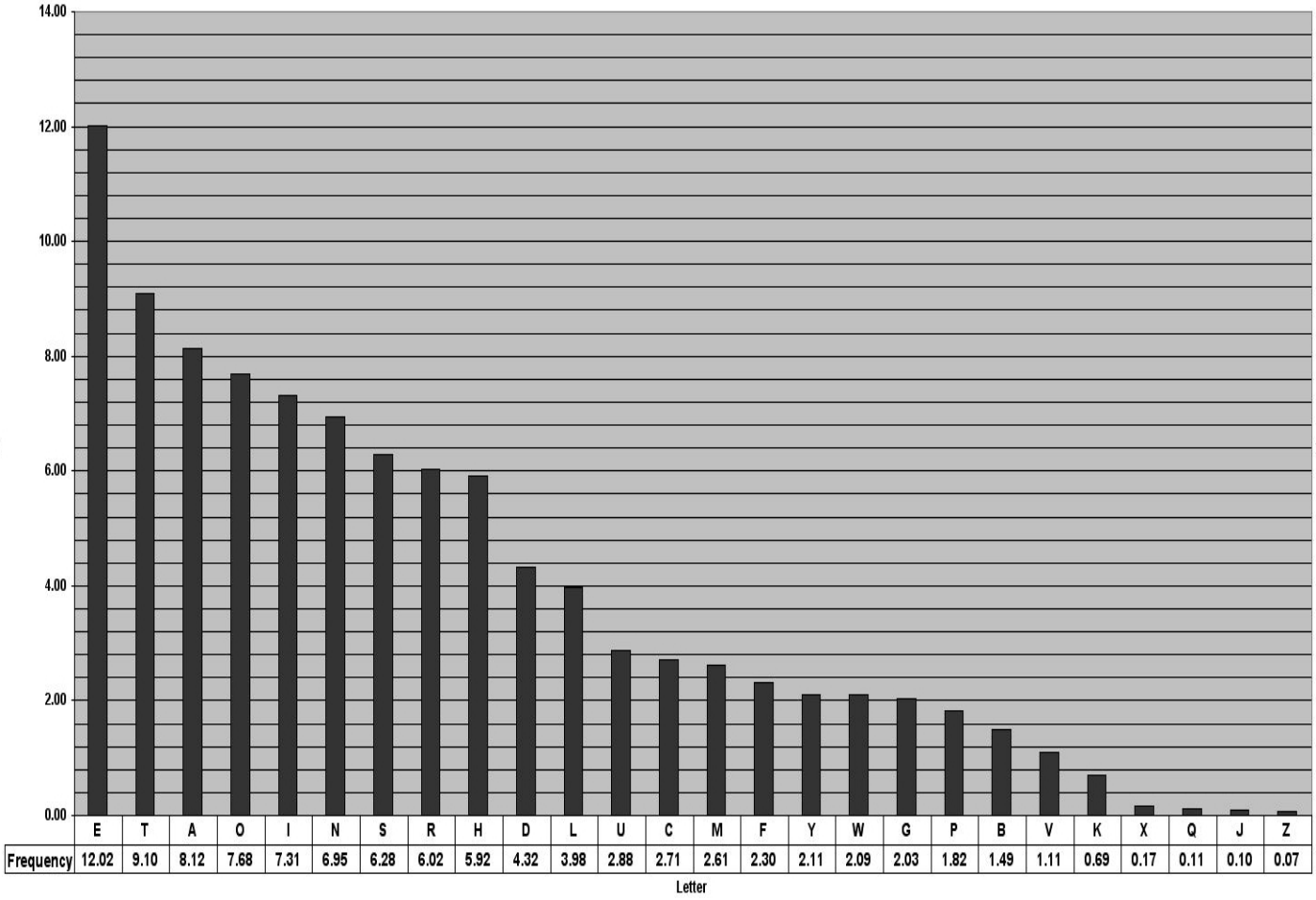
A session key may be established with the following sequence of steps (Figure 14.5).

1. A issues a request to B for a session key and includes a nonce, N_1.
2. B responds with a message that is encrypted using the shared master key. The response includes the session key selected by B, an identifier of B, the value $f(N_1)$, and another nonce, N_2.
3. Using the new session key, A returns $f(N_2)$ to B.



Cryptanalysis:

Letter Frequencies (English alphabet):

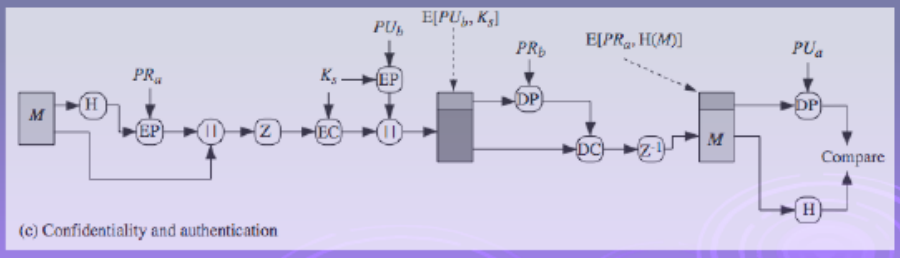


	E	T	A	O	I	N	S	R	H	D	L	U	C	M	F	Y	W	G	P	B	V	K	X	Q	J	Z
Frequency	12.02	9.10	8.12	7.68	7.31	6.95	6.28	6.02	5.92	4.32	3.98	2.88	2.71	2.61	2.30	2.11	2.09	2.03	1.82	1.49	1.11	0.69	0.17	0.11	0.10	0.07

The Kasiski/Kerckhoff Method:

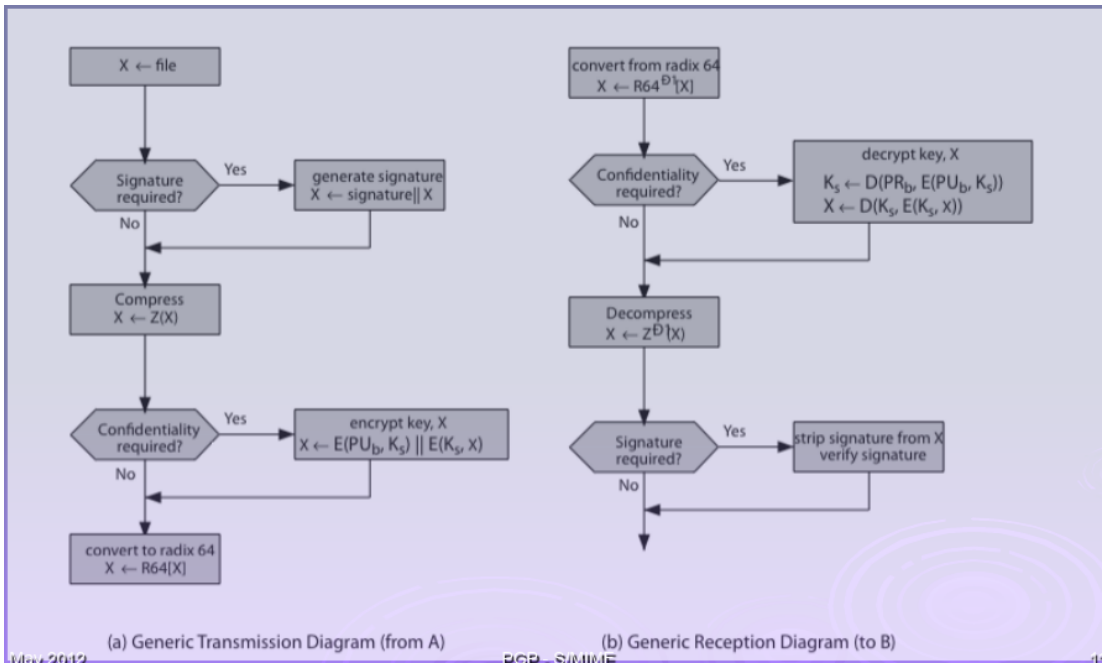
PGP/MIME EMAIL SECURITY:

- can use both services on same message
  - create signature & attach to message
  - encrypt both message & signature
  - attach RSA/ElGamal encrypted session key



(c) Confidentiality and authentication

- when using PGP will have binary data to send (encrypted message etc.)
- however email was designed only for text
- hence PGP must encode raw binary data into printable ASCII characters
- uses radix-64 algorithm (aka "ASCII Armour")
  - maps 3 bytes to 4 printable chars (it's the Base64 of MIME)
  - also appends a 24-bit CRC
- PGP also segments messages if too big



(a) Generic Transmission Diagram (from A)

(b) Generic Reception Diagram (to B)

Birthday Attack:

We consider the following experiment. From a set of *H* values we choose *n* values uniformly at random thereby allowing repetitions. Let *p*(*n*;*H*) be the probability that during this experiment at least one value is chosen more than once. This probability can be approximated as

Let *n*(*p*;*H*) be the smallest number of values we have to choose, such that the probability for finding a collision is at least *p*. By inverting this expression above, we find the following approximation

$$p(n;H) \approx 1 - e^{-n(n-1)/(2H)} \approx 1 - e^{-n^2/(2H)},$$

e^-x = 1 – x, x << 1

$$p(n) \approx \frac{n^2}{2H}$$

which can also be written as

$$H \approx \frac{n^2}{2p(n)}.$$

or

$$n \approx \sqrt{2H \times p(n)}.$$

Proof : The probability $P[\exists i \neq j : k_i = k_j] = 1 - P[\forall i \neq j : k_i \neq k_j]$. In other words, we find the probability such that no collision occurs among the *n* samples we choose and subtract this from 1. Hence,

$P[\exists i \neq j : k_i = k_j] = 1 - (\frac{x}{x})(\frac{x-1}{x})(\frac{x-2}{x})...(\frac{x-(n-1)}{x})$
$= 1 - \prod_{i=1}^{n-1}(1 - \frac{i}{x})...(*)$

Firewalls:

- **In order:** The firewall rules are processed top to bottom. The rule that matches the current IP packet is used. The remaining rules in the list are not considered. The administrator should take care when specifying the correct order of the rules. An incorrect order can have drastically different results.
- **Deny first:** Firewall rules that explicitly deny certain packets are processed first. A matching rule blocks the current IP packet. If no Deny rule matches, the Allow rules are processed next.
- **Best fit:** The firewall uses its own methods to determine the order in which the list of firewall rules is processed, which usually means going from detailed rules to general rules.

Client Packet Type	NMap Command	Port Policy	Response	Inferred Port State
TCP	nmap [-sT | -sS] -Pn <server>	Accept	TCP SYN/ACK	Open
TCP	nmap [-sT | -sS] -Pn <server>	Drop	(none)	Filtered
TCP	nmap [-sT | -sS] -Pn <server>	Reject	TCP RESET	Closed
UDP	nmap -sU -Pn <server>	Accept	(none)	Open or Filtered
UDP	nmap -sU -Pn <server>	Drop	(none)	Open or Filtered
UDP	nmap -sU -Pn <server>	Reject	ICMP Port Unreachable	Closed

General Notes:

Diffie-Hellman:

- Diffie-Hellman can be only used to counter Passive attacks. A Man-in-the-middle can intercept the A being sent from Alice to Bob, generate his own g ^ e then send it to Bob. And again intercept B being sent from Bob to Alice, and send g ^ e to Alice. Now only Eve can understand the messages sent to/from Alice and Bob. Thus Active attacks are possible
- Alice and Bob use the Diffie-Hellman algorithm to exchange a secret key. Eve intercepts the following values: {p = 283, g = 12, A = 77, B = 196 }. Eve computes secret key s by:
  s = B ^ a mod p **OR** s = A ^ b mod p.
  Find a such as g ^ a (mod p) = A
  12 ^ 4 mod 283 = 77 => a = 4
  Therefore s = 196 ^ 4 (mod 283) = 90

RSA:

- Alice selects 2 primes {p = 5, q = 11}.
  n = pq = 55
  φ(n) = (p – 1)(q – 1) = 40
  She selects public key exponent e = 3
  GCD(3, 40) = 1 => Valid selection
  Private exponent d is derived as:
  de = 1 mod φ(n) <=> 3d = 1 mod 40 <=> d = 27
  We want to send M=4 to Alice:
  C = M ^ e (mod n) = 4 ^ 3 (mod 55) = 9
  Alice receives:
  M = C ^ d (mod n) = 9 ^ 27 (mod 55) = 4
- Digital Signature with RSA:
  Alice publishes {n = pq = 221, e = 13}
  Bob receives {M = 65, S = 182} and verifies the signature as following:
  Signature valid if: M = S ^ e (mod n)