

Confidentiality Using Symmetric Encryption

Dr. Arjan Duresi
Louisiana State University
Baton Rouge, LA 70810
Duresi@csc.lsu.edu

These slides are available at:

<http://www.csc.lsu.edu/~duresi/csc4601-07/>

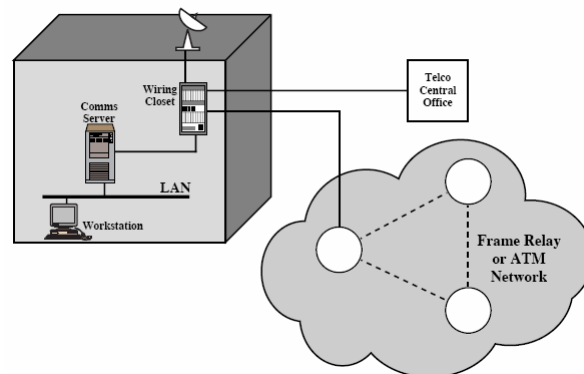


- Use of symmetric encryption to protect confidentiality
- Need for good key distribution
- Use of trusted third party KDC's
- Random number generation

Confidentiality using Symmetric Encryption

- ❑ Traditionally symmetric encryption is used to provide message confidentiality
- ❑ Consider typical scenario
 - workstations on LANs access other workstations & servers on LAN
 - LANs interconnected using switches/routers
 - with external lines or radio/satellite links
- ❑ Consider attacks and placement in this scenario
 - snooping from another workstation
 - use dial-in to LAN or server to snoop
 - use external router link to enter & snoop
 - monitor and/or modify traffic on external links

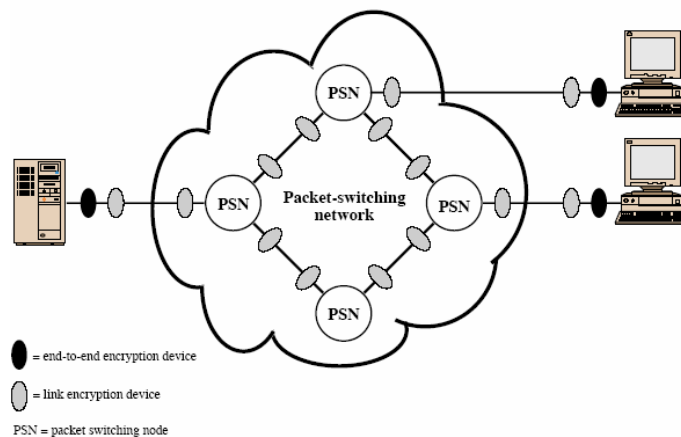
Confidentiality using Symmetric Encryption



Confidentiality using Symmetric Encryption

- ❑ Have two major placement alternatives
- ❑ **link encryption**
 - encryption occurs independently on every link
 - implies must decrypt traffic between links
 - requires many devices, but paired keys
- ❑ **end-to-end encryption**
 - encryption occurs between original source and final destination
 - need devices at each end with shared keys

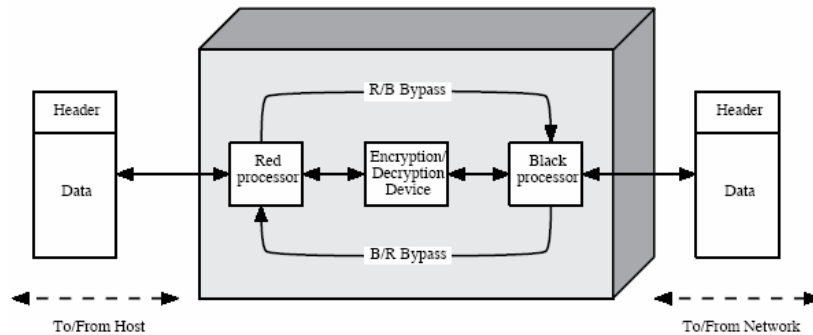
Confidentiality using Symmetric Encryption



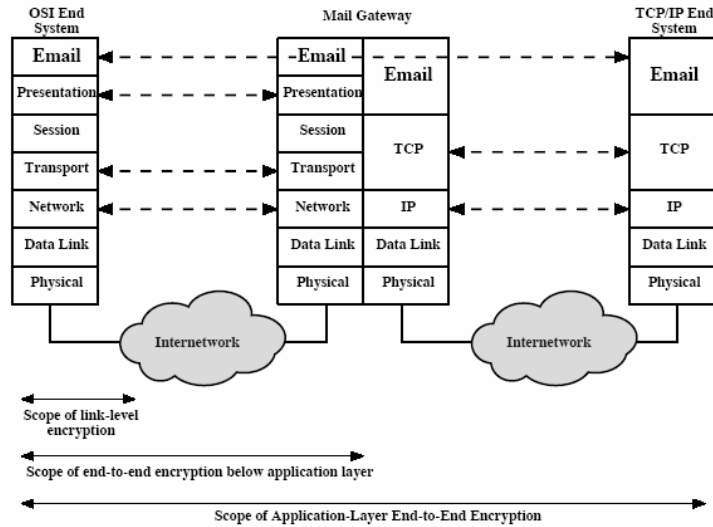
Link vs. End-to-End

Link Encryption	End-to-End Encryption
<i>Security within End Systems and Intermediate Systems</i>	
Message exposed in sending host	Message encrypted in sending host
Message exposed in intermediate nodes	Message encrypted in intermediate nodes
<i>Role of User</i>	
Applied by sending host	Applied by sending process
Transparent to user	User applies encryption
Host maintains encryption facility	User must determine algorithm
One facility for all users	Users selects encryption scheme
Can be done in hardware	Software implementation
All or no messages encrypted	User chooses to encrypt, or not, for each message
<i>Implementation Concerns</i>	
Requires one key per (host-intermediate node) pair and (intermediate node-intermediate node) pair	Requires one key per user pair
Provides host authentication	Provides user authentication

Confidentiality using Symmetric Encryption



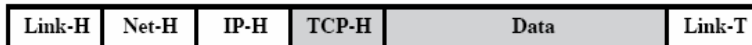
End-to-End



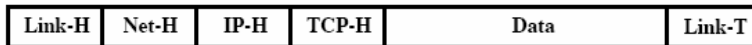
End-to-End



(a) Application-Level Encryption (on links and at routers and gateways)



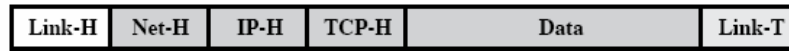
On links and at routers



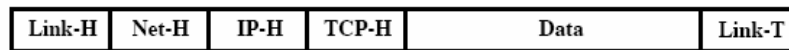
In gateways

(b) TCP-Level Encryption

Link



On links



In routers and gateways

(c) **Link-Level Encryption**

Traffic Confidentiality using Symmetric Encryption Analysis

- ❑ When using end-to-end encryption must leave headers in clear
 - so network can correctly route information
- ❑ hence although contents protected, traffic pattern flows are not
- ❑ ideally want both at once
 - end-to-end protects data contents over entire path and provides authentication
 - link protects traffic flows from monitoring

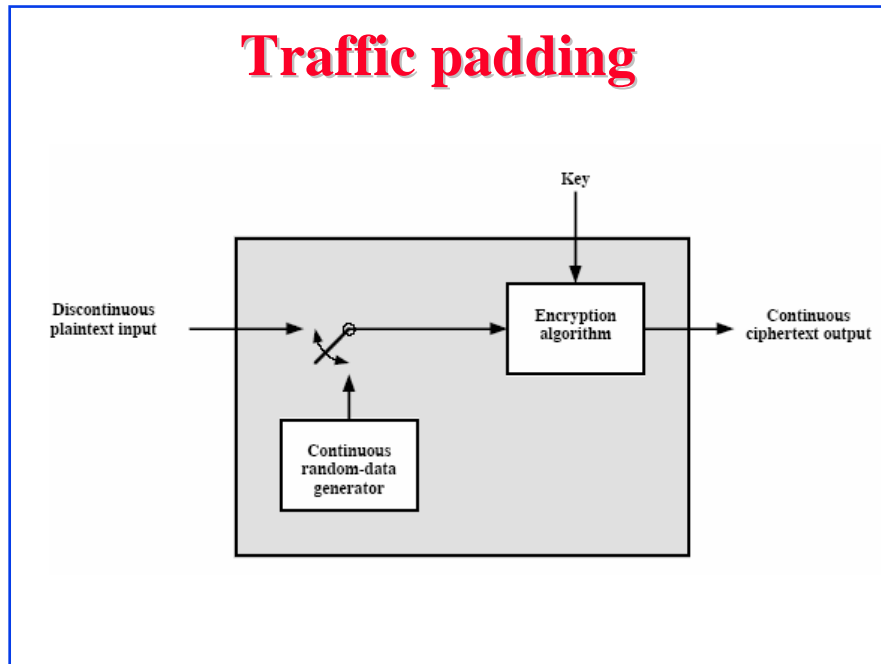
Placement of Encryption

- ❑ Can place encryption function at various layers in OSI Reference Model
 - link encryption occurs at layers 1 or 2
 - end-to-end can occur at layers 3, 4, 6, 7
 - as move higher less information is encrypted but it is more secure though more complex with more entities and keys

Traffic Analysis

- ❑ Is monitoring of communications flows between parties
 - useful both in military & commercial spheres
 - can also be used to create a covert channel
- ❑ link encryption obscures header details
 - but overall traffic volumes in networks and at end-points is still visible
- ❑ traffic padding can further obscure flows
 - but at cost of continuous traffic

Traffic padding



Louisiana State University 8- Confidentiality using Symmetric Encryption - 15 CSC4601 S07

Key Distribution

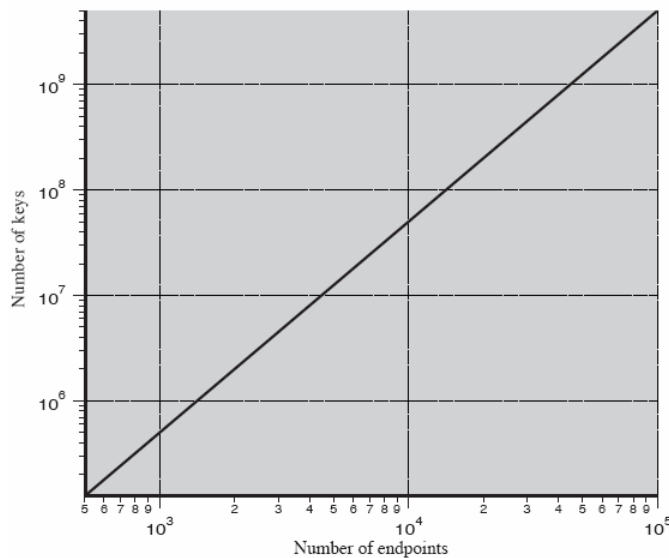
- Symmetric schemes require both parties to share a common secret key
- Issue is how to securely distribute this key
- Often secure system failure due to a break in the key distribution scheme

Louisiana State University 8- Confidentiality using Symmetric Encryption - 16 CSC4601 S07

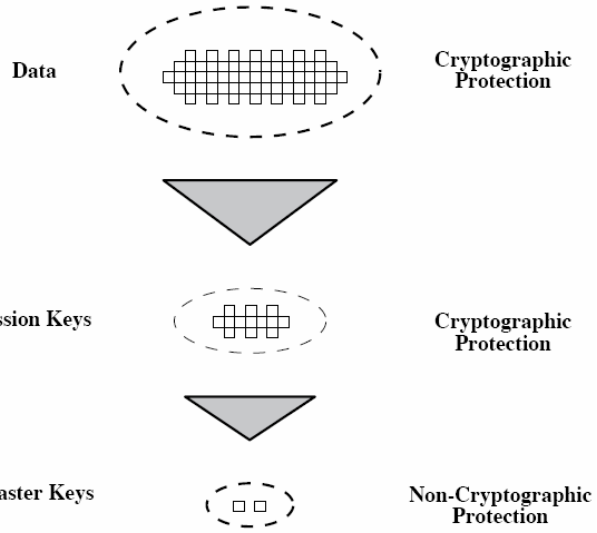
Key Distribution

- Given parties A and B have various **key distribution** alternatives:
 1. A can select key and physically deliver to B
 2. third party can select & deliver key to A & B
 3. if A & B have communicated previously can use previous key to encrypt a new key
 4. if A & B have secure communications with a third party C, C can relay key between A & B

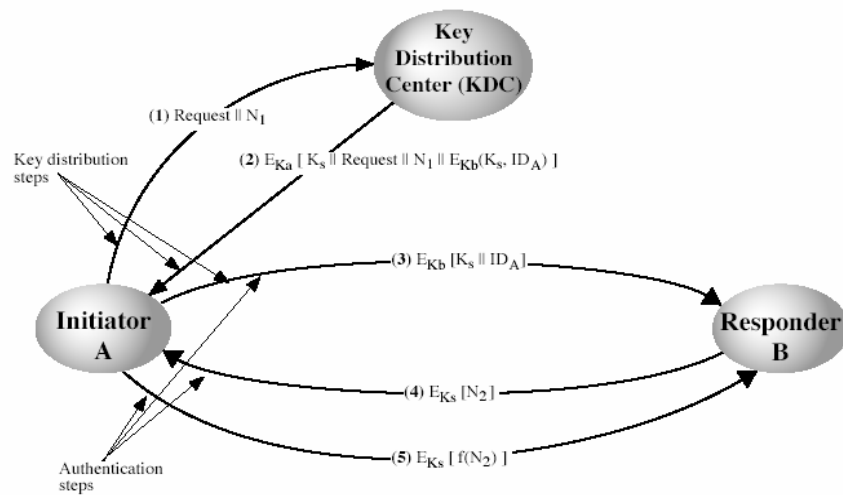
Key Distribution



Use of Key Hierarchy



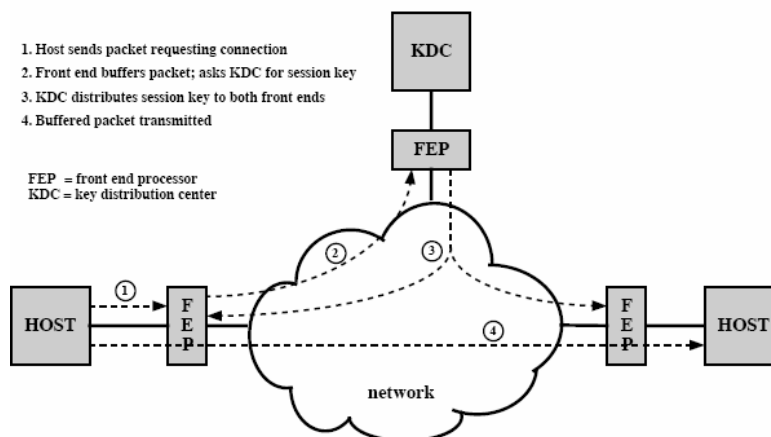
Key Distribution Scenario



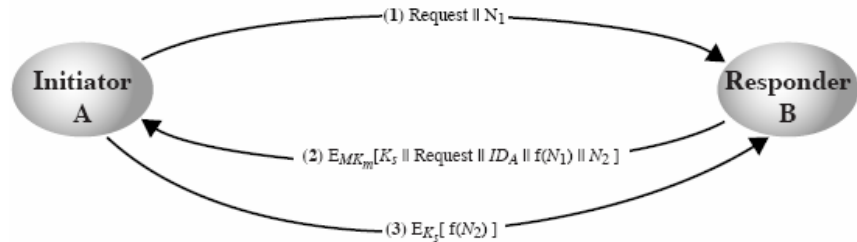
Key Distribution Issues

- ❑ Hierarchies of KDC's required for large networks, but must trust each other
- ❑ Session key lifetimes should be limited for greater security
- ❑ use of automatic key distribution on behalf of users, but must trust system
- ❑ use of decentralized key distribution
- ❑ controlling purposes keys are used for

Automatic Key Distribution



Decentralized Key Distribution



- Each Node maintains (n-1) master keys
- Messages transmitted using the master key are short
⇒ Cryptanalysis is difficult

Controlling Key Usage

- Key hierarchy
- Define different types of session keys on the basis of their use:
 - Data Encryption key – for general communications
 - PIN-encrypting key – for personal identification, used in electronic funds transfer
 - File-encryption key – for encrypting files stored in public domains

Random Numbers

- ❑ Many uses of **random numbers** in cryptography
 - nonces in authentication protocols to prevent replay
 - session keys
 - public key generation
 - keystream for a one-time pad
- ❑ In all cases its critical that these values be
 - statistically random
 - ❑ with uniform distribution, independent
 - unpredictable cannot infer future sequence on previous values

Natural Random Noise

- ❑ Best source is natural randomness in real world
- ❑ Find a regular but random event and monitor
- ❑ do generally need special h/w to do this
 - eg. radiation counters, radio noise, audio noise, thermal noise in diodes, leaky capacitors, mercury discharge tubes etc
- ❑ starting to see such h/w in new CPU's
- ❑ problems of **bias** or uneven distribution in signal
 - have to compensate for this when sample and use
 - best to only use a few noisiest bits from each sample

Published Sources

- ❑ A few published collections of random numbers
- ❑ Rand Co, in 1955, published 1 million numbers
 - generated using an electronic roulette wheel
 - has been used in some cipher designs of Khafre
- ❑ Earlier Tippett in 1927 published a collection
- ❑ Issues are that:
 - these are limited
 - too well-known for most uses

Pseudorandom Number Generators (PRNGs)

- ❑ Algorithmic technique to create “random numbers”
 - although not truly random
 - can pass many tests of “randomness”

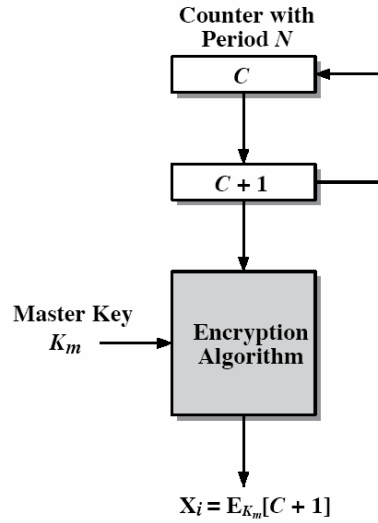
Linear Congruential Generator

- ❑ Common iterative technique using:
$$X_{n+1} = (aX_n + c) \bmod m$$
- ❑ If a, c, m, X_0 are integers, it will produce a sequence of integers with each integer in the range $0 \leq X_n \leq m$.
- ❑ The selection of a, c, m is critical in developing a good random number generator.
 - m - large as possible, near 2^{31} is typically chosen.
 - If it can be shown that if m is prime and $c=0$, then for certain values of a , the period of the generating function is $m-1$.
- ❑ Given suitable values of parameters can produce a long random-like sequence
- ❑ Suitable criteria to have are:
 - function generates a full-period
 - generated sequence should appear random
 - efficient implementation with 32-bit arithmetic
- ❑ Among more than 2 billions of possible choices for a , only few of them pass all three tests.
- ❑ Note that an attacker can reconstruct sequence given a small number of values

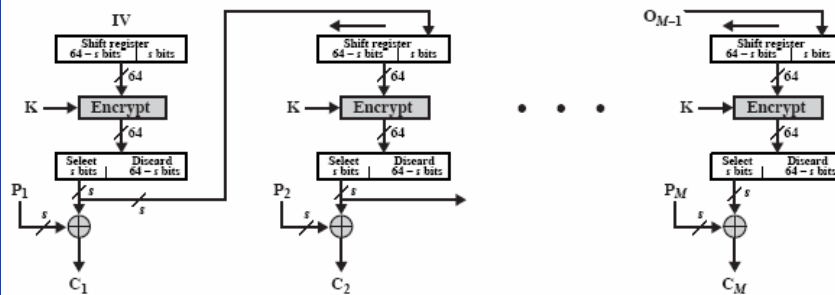
Using Block Ciphers as Stream Ciphers

- ❑ Can use block cipher to generate numbers
- ❑ Use Counter Mode
$$X_i = E_{Km}[C_i]$$
- ❑ Use Output Feedback Mode
$$X_i = E_{Km}[X_{i-1}]$$
- ❑ ANSI X9.17 PRNG
 - uses date-time + seed inputs and 3 triple-DES encryptions to generate new seed & random

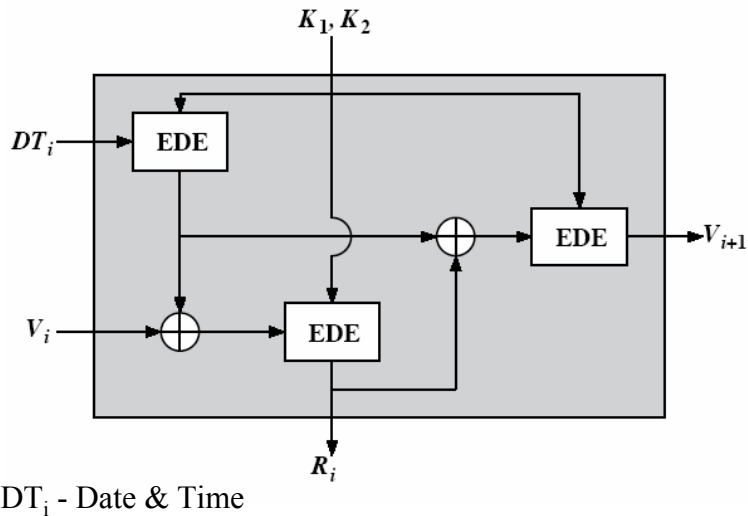
Pseudorandom Number Generation from a Counter



Use Output Feedback Mode



ANSI X9.17 PRNG



Blum Blum Shub Generator

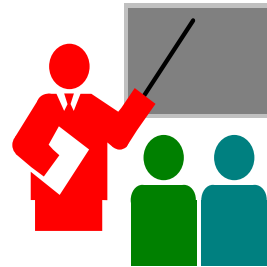
- Based on public key algorithms
- Use least significant bit from iterative equation:
 - $x_{i+1} = x_i^2 \pmod n$
 - $B_i = x_i \pmod 2$
 - where $n=p \cdot q$, and primes $p=q=3 \pmod 4$
 - s prime to n
- Unpredictable, passes **next-bit** test
 - If there is not a polynomial-time algorithm that, on input of the first k bits of an output sequence, can predict the $(k+1)^{st}$ with probability significantly more than $\frac{1}{2}$.
- Security rests on difficulty of factoring n
- Is unpredictable given any run of bits
- Slow, since very large numbers must be used
- Too slow for cipher use, good for key generation

BBS

s	X_i	B_i
0	20749	
1	143135	1
2	177671	1
3	97048	0
4	89992	0
5	174051	1
6	80649	1
7	45663	1
8	69442	0
9	186894	0
10	177046	0

s	X_i	B_i
11	137922	0
12	123175	1
13	8630	0
14	114386	0
15	14863	1
16	133015	1
17	106065	1
18	45870	0
19	137171	1
20	48060	0

Summary



- Have considered:
 - use of symmetric encryption to protect confidentiality
 - need for good key distribution
 - use of trusted third party KDC's
 - random number generation