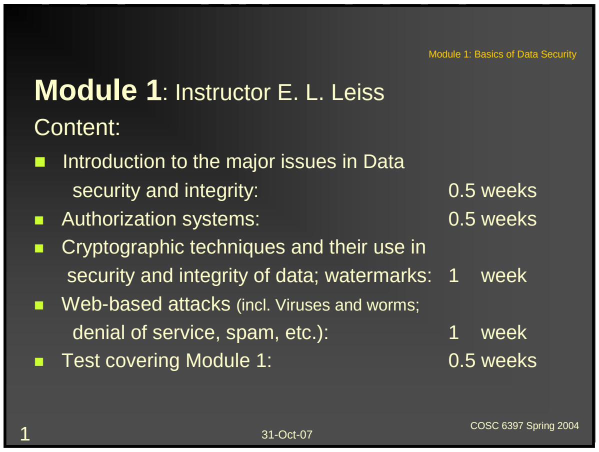
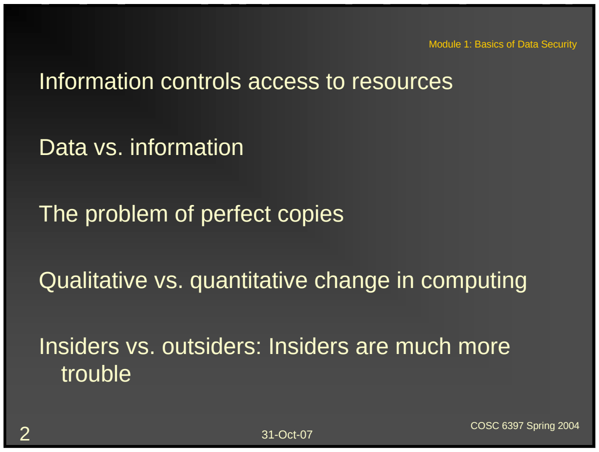


Module 1: Instructor E. L. Leiss

Content:

- Introduction to the major issues in Data security and integrity: 0.5 weeks
- Authorization systems: 0.5 weeks
- Cryptographic techniques and their use in security and integrity of data; watermarks: 1 week
- Web-based attacks (incl. Viruses and worms; denial of service, spam, etc.): 1 week
- Test covering Module 1: 0.5 weeks



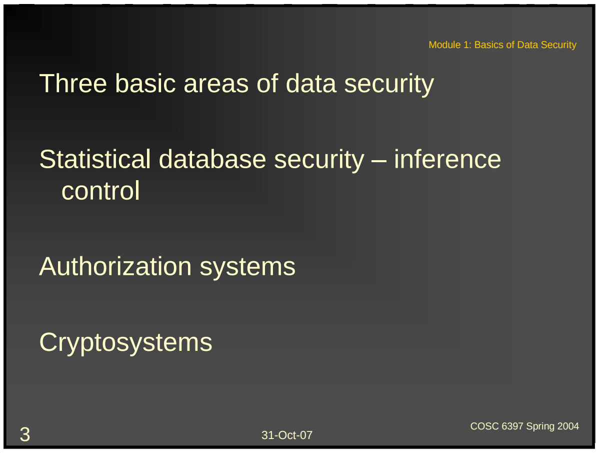
Information controls access to resources

Data vs. information

The problem of perfect copies

Qualitative vs. quantitative change in computing

Insiders vs. outsiders: Insiders are much more trouble

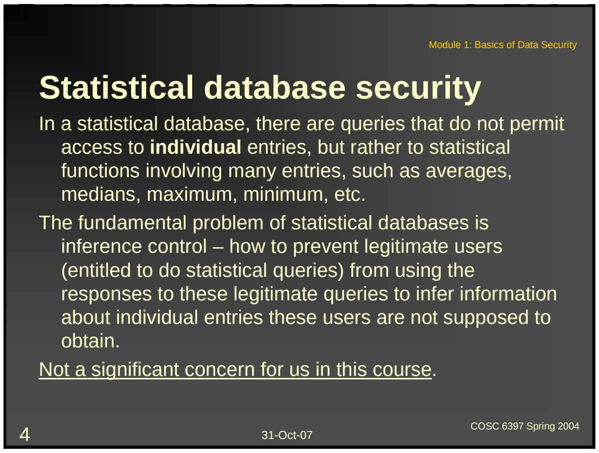


Three basic areas of data security

Statistical database security – inference control

Authorization systems

Cryptosystems

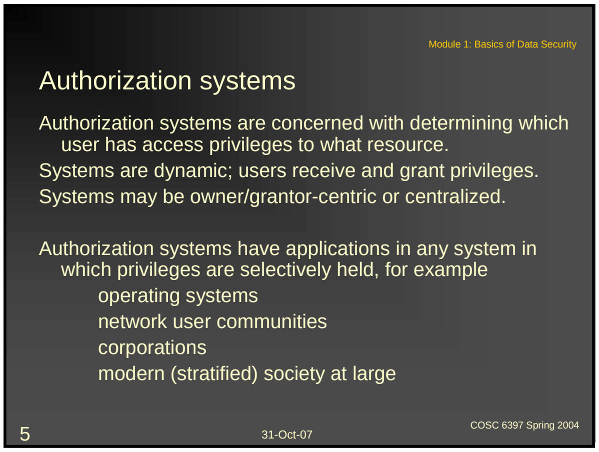


Statistical database security

In a statistical database, there are queries that do not permit access to **individual** entries, but rather to statistical functions involving many entries, such as averages, medians, maximum, minimum, etc.

The fundamental problem of statistical databases is inference control – how to prevent legitimate users (entitled to do statistical queries) from using the responses to these legitimate queries to infer information about individual entries these users are not supposed to obtain.

Not a significant concern for us in this course.



Authorization systems

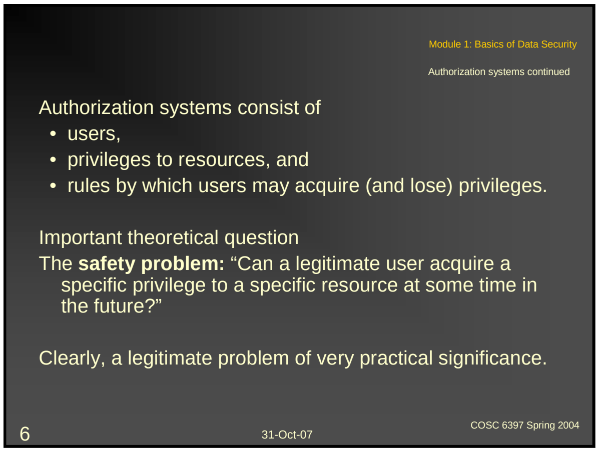
Authorization systems are concerned with determining which user has access privileges to what resource.

Systems are dynamic; users receive and grant privileges.

Systems may be owner/grantor-centric or centralized.

Authorization systems have applications in any system in which privileges are selectively held, for example

operating systems
network user communities
corporations
modern (stratified) society at large



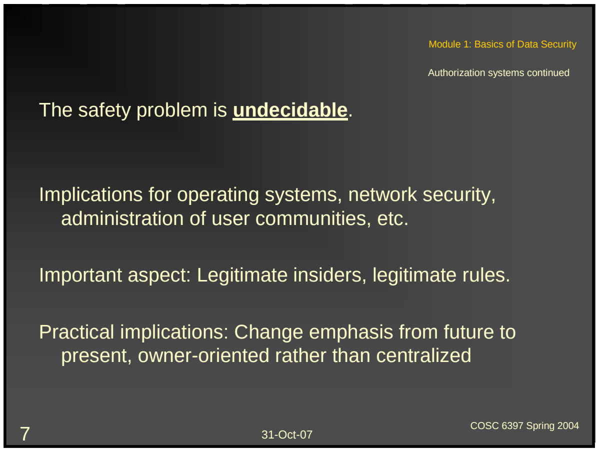
Authorization systems consist of

- users,
- privileges to resources, and
- rules by which users may acquire (and lose) privileges.

Important theoretical question

The **safety problem:** “Can a legitimate user acquire a specific privilege to a specific resource at some time in the future?”

Clearly, a legitimate problem of very practical significance.

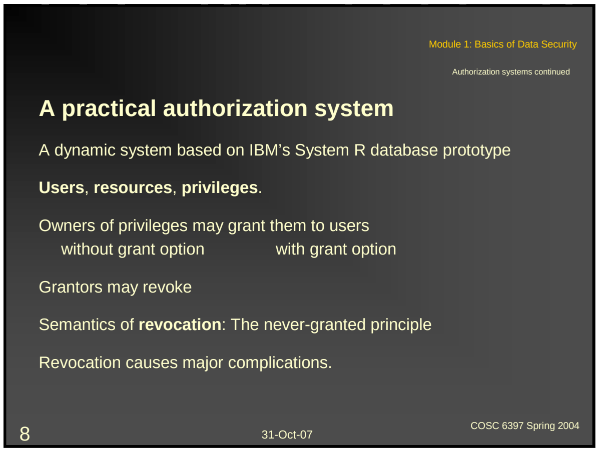


The safety problem is **undecidable**.

Implications for operating systems, network security, administration of user communities, etc.

Important aspect: Legitimate insiders, legitimate rules.

Practical implications: Change emphasis from future to present, owner-oriented rather than centralized



A practical authorization system

A dynamic system based on IBM's System R database prototype

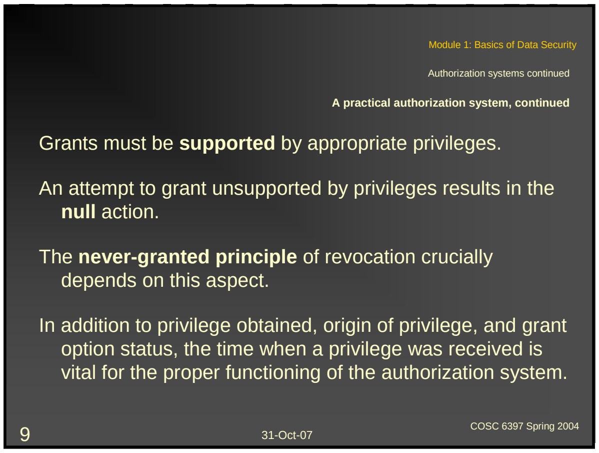
Users, **resources**, **privileges**.

Owners of privileges may grant them to users
without grant option with grant option

Grantors may revoke

Semantics of **revocation**: The never-granted principle

Revocation causes major complications.

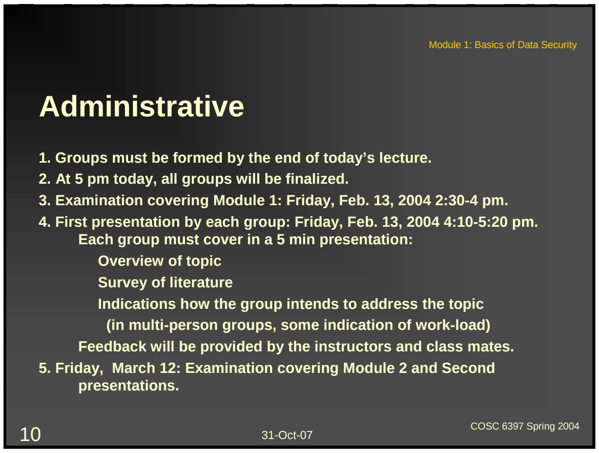


Grants must be **supported** by appropriate privileges.

An attempt to grant unsupported by privileges results in the **null** action.

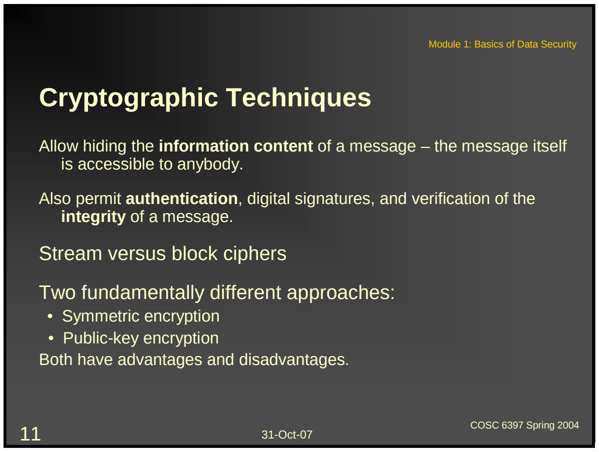
The **never-granted principle** of revocation crucially depends on this aspect.

In addition to privilege obtained, origin of privilege, and grant option status, the time when a privilege was received is vital for the proper functioning of the authorization system.



Administrative

1. Groups must be formed by the end of today's lecture.
2. At 5 pm today, all groups will be finalized.
3. Examination covering Module 1: Friday, Feb. 13, 2004 2:30-4 pm.
4. First presentation by each group: Friday, Feb. 13, 2004 4:10-5:20 pm.
Each group must cover in a 5 min presentation:
Overview of topic
Survey of literature
Indications how the group intends to address the topic
(in multi-person groups, some indication of work-load)
Feedback will be provided by the instructors and class mates.
5. Friday, March 12: Examination covering Module 2 and Second presentations.



Cryptographic Techniques

Allow hiding the **information content** of a message – the message itself is accessible to anybody.

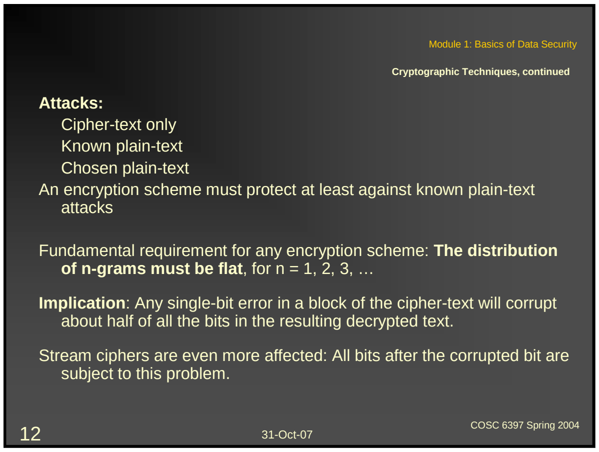
Also permit **authentication**, digital signatures, and verification of the **integrity** of a message.

Stream versus block ciphers

Two fundamentally different approaches:

- Symmetric encryption
- Public-key encryption

Both have advantages and disadvantages.

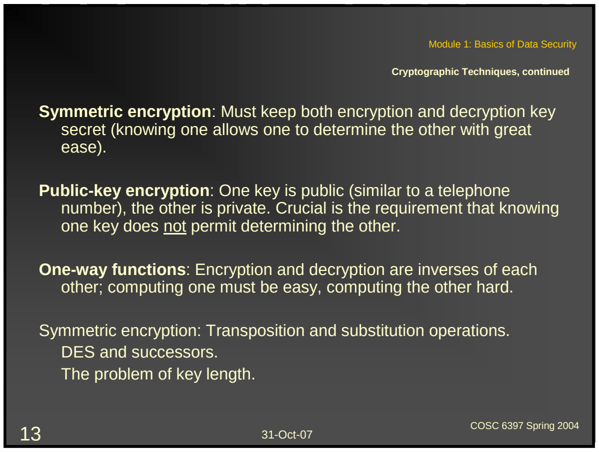


Attacks:
Cipher-text only
Known plain-text
Chosen plain-text
An encryption scheme must protect at least against known plain-text attacks

Fundamental requirement for any encryption scheme: **The distribution of n-grams must be flat**, for n = 1, 2, 3, …

Implication: Any single-bit error in a block of the cipher-text will corrupt about half of all the bits in the resulting decrypted text.

Stream ciphers are even more affected: All bits after the corrupted bit are subject to this problem.



Cryptographic Techniques, continued

Symmetric encryption: Must keep both encryption and decryption key secret (knowing one allows one to determine the other with great ease).

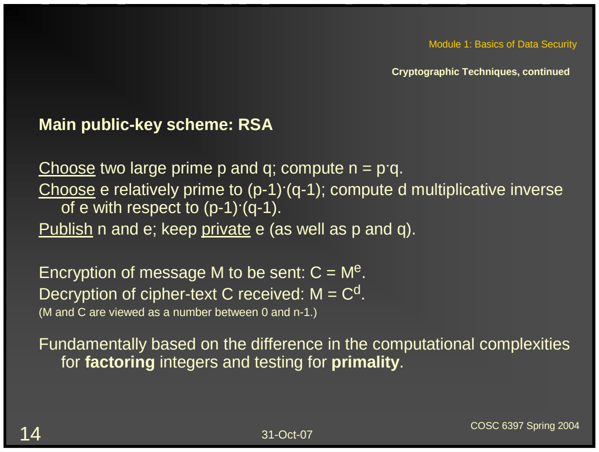
Public-key encryption: One key is public (similar to a telephone number), the other is private. Crucial is the requirement that knowing one key does <u>not</u> permit determining the other.

One-way functions: Encryption and decryption are inverses of each other; computing one must be easy, computing the other hard.

Symmetric encryption: Transposition and substitution operations.

DES and successors.

The problem of key length.



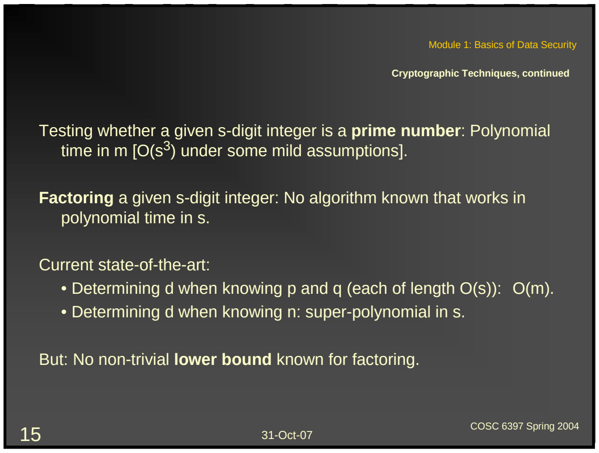
Cryptographic Techniques, continued

Main public-key scheme: RSA

<u>Choose</u> two large prime p and q; compute n = p·q.
<u>Choose</u> e relatively prime to (p-1)·(q-1); compute d multiplicative inverse of e with respect to (p-1)·(q-1).
<u>Publish</u> n and e; keep <u>private</u> e (as well as p and q).

Encryption of message M to be sent: $C = M^e$.
Decryption of cipher-text C received: $M = C^d$.
(M and C are viewed as a number between 0 and n-1.)

Fundamentally based on the difference in the computational complexities for **factoring** integers and testing for **primality**.



Cryptographic Techniques, continued

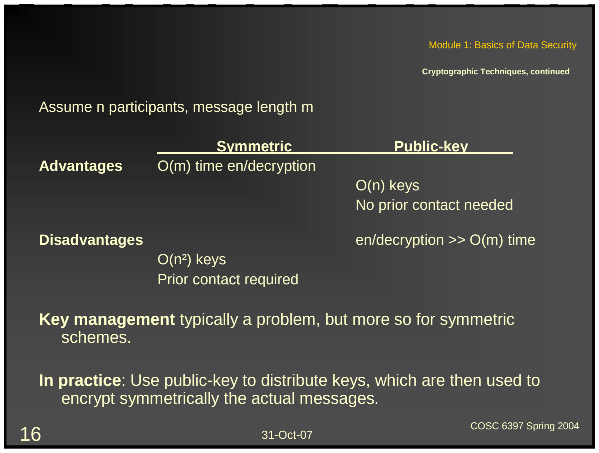
Testing whether a given s-digit integer is a **prime number**: Polynomial time in m [$O(s^3)$ under some mild assumptions].

Factoring a given s-digit integer: No algorithm known that works in polynomial time in s.

Current state-of-the-art:

- Determining d when knowing p and q (each of length O(s)): O(m).
- Determining d when knowing n: super-polynomial in s.

But: No non-trivial **lower bound** known for factoring.



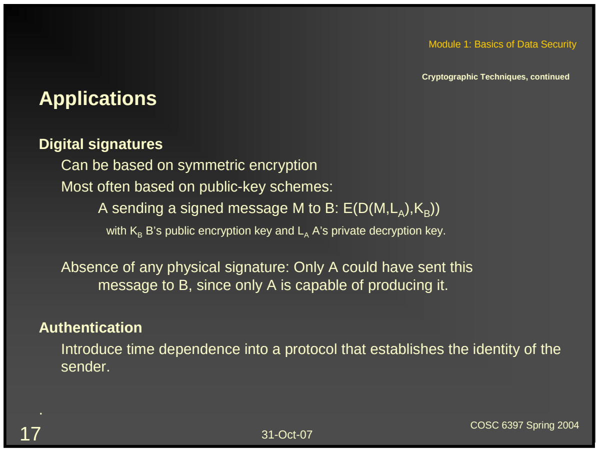
Cryptographic Techniques, continued

Assume n participants, message length m

	Symmetric	**Public-key**
Advantages	O(m) time en/decryption	
		O(n) keys
		No prior contact needed
Disadvantages		en/decryption >> O(m) time
	$O(n^2)$ keys	
	Prior contact required	

Key management typically a problem, but more so for symmetric schemes.

In practice: Use public-key to distribute keys, which are then used to encrypt symmetrically the actual messages.



Cryptographic Techniques, continued

Applications

Digital signatures

Can be based on symmetric encryption

Most often based on public-key schemes:

A sending a signed message M to B: $E(D(M,L_A),K_B)$

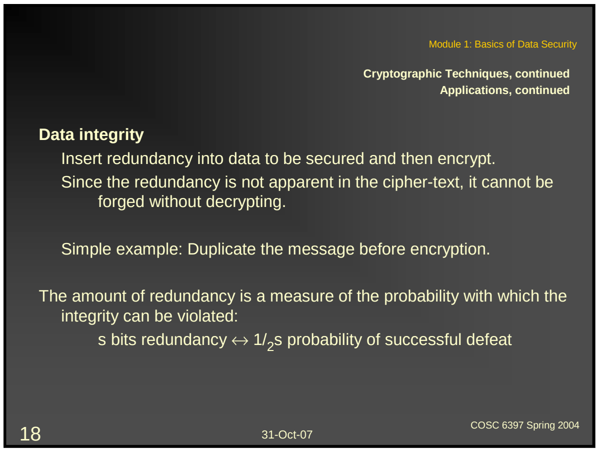
with K_B B's public encryption key and L_A A's private decryption key.

Absence of any physical signature: Only A could have sent this message to B, since only A is capable of producing it.

Authentication

Introduce time dependence into a protocol that establishes the identity of the sender.

.



Cryptographic Techniques, continued
Applications, continued

Data integrity

Insert redundancy into data to be secured and then encrypt.

Since the redundancy is not apparent in the cipher-text, it cannot be forged without decrypting.

Simple example: Duplicate the message before encryption.

The amount of redundancy is a measure of the probability with which the integrity can be violated:

s bits redundancy ↔ $1/_{2}s$ probability of successful defeat