

# Privacy in Databases

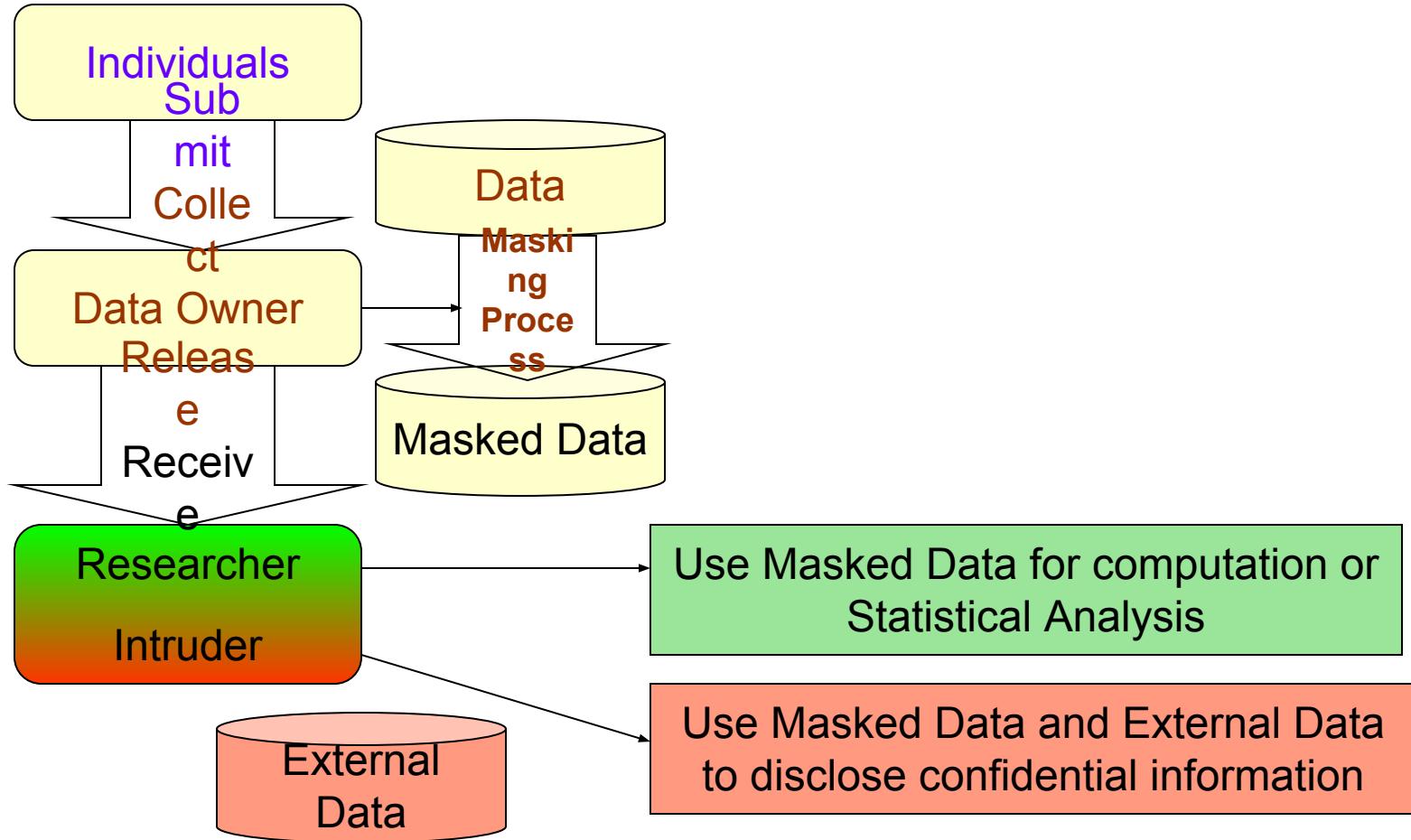
# Introduction to Privacy

- **Privacy** is the control over the extent, timing, and circumstances of sharing oneself (physically, behaviourally, or intellectually) with others.
- Examples of activities considered private might include
  - a medical examination;
  - activities within your home;
  - using a restaurant bathroom;
  - generally any action for which you have the reasonable expectation of privacy.
- Most things done in public places would not be considered private.

# Confidentiality vs. Privacy

- Confidentiality refers to personal information shared with an attorney, physician, therapist,
  - or other individual that generally cannot be divulged to third parties without the express consent of the client.
- On the other hand, privacy refers to the freedom from intrusion into one's personal matters, and personal information.
  - Here, adversary use legitimate methods !!!
- Privacy applies to the person.
- Confidentiality applies to the data.

# Need for privacy



# Some terminologies

- **Data** represents a series of records, each record containing information on an individual unit such as a person, a firm, an institution, etc.
- **Masked data** names and other identifying information are removed from data.
- **External Information** any known information by an presumptive intruder related to some individuals from initial data.

# Need for privacy...

Initial data

Name	SSN	Age	Zip	Diagnosis	Income
Alice	123456789	44	48202	AIDS	17,000
Bob	323232323	44	48202	AIDS	68,000
Charley	232345656	44	48201	Asthma	80,000
Dave	333333333	55	48310	Asthma	55,000
Eva	666666666	55	48310	Diabetes	23,000

Masked data

Age	Zip	Diagnosis	Income
44	48202	AIDS	17,000
44	48202	AIDS	68,000
44	48201	Asthma	80,000
55	48310	Asthma	55,000
55	48310	Diabetes	23,000

Data Owner

# Need for privacy...

Initial Microdata

Name	SSN	Age	Zip	Diagnosis	Income
Alice	123456789	44	48202	AIDS	17,000
Bob	323232323	44	48202	AIDS	68,000
Charley	232345656	44	48201	Asthma	80,000
Dave	333333333	55	48310	Asthma	55,000
Eva	666666666	55	48310	Diabetes	23,000

Data Owner

Masked Microdata

Age	Zip	Diagnosis	Income
44	48202	AIDS	17,000
44	48202	AIDS	68,000
44	48201	Asthma	80,000
55	48310	Asthma	55,000
55	48310	Diabetes	23,000

External Information

Name	SSN	Age	Zip
Alice	123456789	44	48202
Charley	232345656	44	48201
Dave	333333333	55	48310

Intruder

# Need for privacy...

Initial Microdata

Name	SSN	Age	Zip	Diagnosis	Income
Alice	123456789	44	48202	AIDS	17,000
Bob	323232323	44	48202	AIDS	68,000
Charley	232345656	44	48201	Asthma	80,000
Dave	333333333	55	48310	Asthma	55,000
Eva	666666666	55	48310	Diabetes	23,000

Data Owner

Masked Microdata

Age	Zip	Diagnosis	Income
44	48202	AIDS	17,000
44	48202	AIDS	68,000
44	48201	Asthma	80,000
55	48310	Asthma	55,000
55	48310	Diabetes	23,000

Identity Disclosure:

*Charlie is the third record*

Attribute Disclosure:

*Alice has AIDS*

External Information

Name	SSN	Age	Zip
Alice	123456789	44	48202
Charley	232345656	44	48201
Dave	333333333	55	48310

Intruder

# Need for privacy...

Initial data

Name	SSN	Age	Zip	Diagnosis	Income
Alice	123456789	44	48202	AIDS	17,000
Bob	323232323	44	48202	AIDS	68,000
Charley	232345656	44	48201	Asthma	80,000
Dave	333333333	55	48310	Asthma	55,000
Eva	666666666	55	48310	Diabetes	23,000

Data Owner

Masked data

Age	Zip	Diagnosis	Income
44	482	AIDS	17,000
44	482	AIDS	68,000
44	482	Asthma	80,000
55	483	Asthma	55,000
55	483	Diabetes	23,000

Identity Disclosure:

*Charlie is the third record*

Attribute Disclosure:

*Alice has AIDS*

External Information

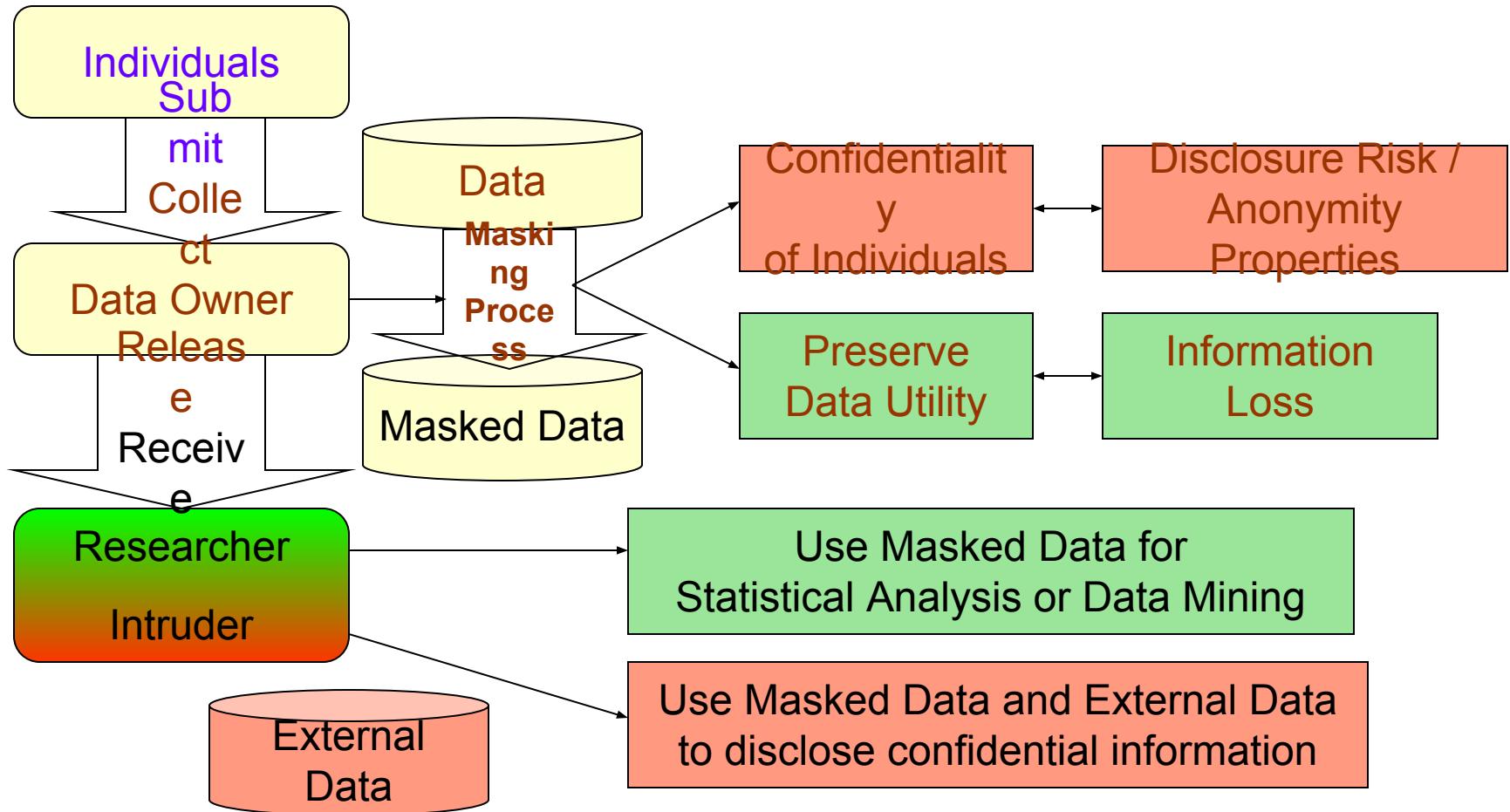
Name	SSN	Age	Zip
Alice	123456789	44	48202
Charley	232345656	44	48201
Dave	333333333	55	48310

Intruder

# Need for privacy...

- Disclosure Risk and Information Loss
  - **Disclosure risk** - the risk that a given form of disclosure will arise if a masked data is released  
[Chen, 1998].
  - **Information loss** - the quantity of information which exist in the initial data and because of disclosure control methods does not occur in masked data  
[Willemborg, 2001].

# Need for privacy...



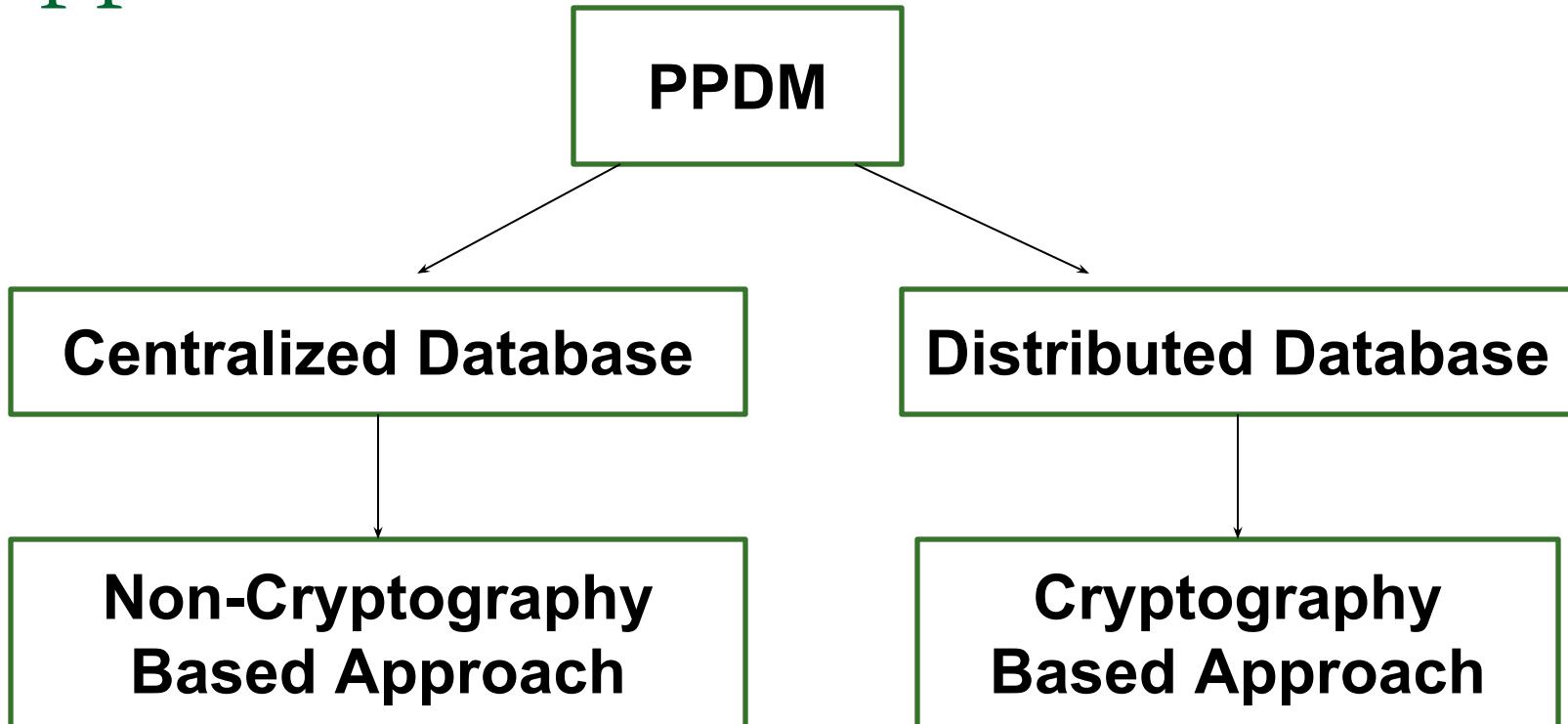
# Privacy-Preserving Data Mining

- Privacy Preserving Data Mining (PPDM)
  - A research direction in data mining and statistical databases, where data mining algorithms are analyzed for the side-effects they incur in data privacy [Verykios 2004].
  - The goal is to get **accurate data mining results** while preserving **privacy**
- Two objectives:
  - Data Privacy
  - Privacy of Data Mining results

# Privacy-Preserving Data Mining...

- Two scenarios
  - Centralized Database
    - Data Mining is performed on the data located at single site
  - Distributed Database
    - Data located at different sites
    - Bringing data together at one place for mining is not possible due to privacy laws or policies
    - Privacy Preserving Distributed Data Mining (PPDDM)

# Privacy-Preserving Data Mining: Approaches



# Privacy-Preserving Data Mining: Approaches...

- State-of-the-art in Privacy Preservation in data mining

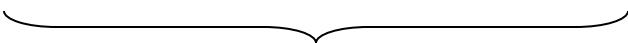
Sr. No.	PPDM Approaches	Privacy	Overheads	Remarks
1	Cryptography based	High	High	Poor Scalability
2	Non-Cryptography based	Low	Low	Privacy-Accuracy trade-off

# Non-Cryptography based approaches

# Source of problem

- Even if we remove the direct uniquely identifying attributes
  - There are some fields that may still uniquely identify some individual!
  - The attacker can *join* them with other sources and identify individuals

	Non-Sensitive Data				Sensitive Data
#	Zip	Age	Nationality	Condition	
...	...	...	...	...	...

  
**Quasi-Identifiers**

# K-anonymity

- Proposed by Sweeney
- Change data in such a way that for each tuple in the resulting table there are atleast  $(k-1)$  other tuples with the same value for the quasi-identifier – **K-anonymized table**

#	<i>Zip</i>	<i>Age</i>	<i>Nationality</i>	<i>Condition</i>
1	130**	< 40	*	Heart Disease
2	130**	< 40	*	Heart Disease
3	130**	< 40	*	Viral Infection
4	130**	< 40	*	Cancer

4-anonymized

# Techniques for anonymization

- Data Swapping
- Randomization
- Generalization
  - Replace the original value by a semantically consistent but less specific value
- Suppression
  - Data not released at all

# Techniques for anonymization...

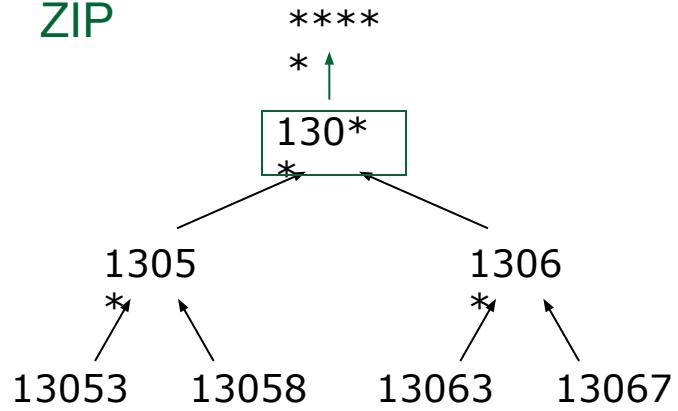
#	Zip	Age	Nationality	Condition
1	130**	< 40	*	Heart Disease
2	130**	< 40	*	Heart Disease
3	130**	< 40	*	Viral Infection
4	130**	< 40	*	Cancer

Generalization

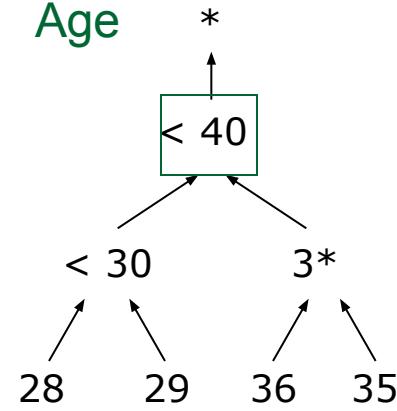
Suppression

# Generalization hierarchies

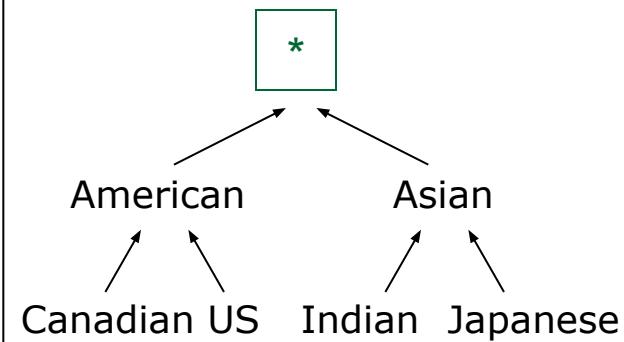
ZIP



Age



Nationality



- **Generalization Hierarchies:** Data owner defines how values can be generalized
- **Table Generalization:** A table generalization is created by generalizing all values in a column to a specific level of generalization

# K-minimal generalizations

- There are many k-anonymizations – which *one* to pick?
  - Intuition: The one that does not generalize the data more than needed (decrease in utility of the published dataset!)
- K-minimal generalization
  - A k-anonymized table that is not a generalization of another k-anonymized table

#	<b>Zip</b>	<b>Age</b>	<b>Nationality</b>	<b>Condition</b>
1	13053	< 40	*	Heart Disease
2	13053	< 40	*	Viral Infection
3	13067	< 40	*	Heart Disease
4	13067	< 40	*	Cancer

2-minimal  
Generalizations

#	<b>Zip</b>	<b>Age</b>	<b>Nationality</b>	<b>Condition</b>
1	130**	< 30	American	Heart Disease
2	130**	< 30	American	Viral Infection
3	130**	3*	Asian	Heart Disease
4	130**	3*	Asian	Cancer

#	<b>Zip</b>	<b>Age</b>	<b>Nationality</b>	<b>Condition</b>
1	130**	< 40	*	Heart Disease
2	130**	< 40	*	Viral Infection
3	130**	< 40	*	Heart Disease
4	130**	< 40	*	Cancer

NOT a  
2-minimal  
Generalization

# K-Anonymity drawbacks

- K-anonymity alone *does not* provide full privacy!
- Suppose attacker knows the non-sensitive attributes of

	<i>Zip</i>	<i>Age</i>	<i>National</i>
Bob →	13053	31	American
Umeko →	13068	21	Japanese

- And the fact that Japanese have very low incidence of heart disease

# K-Anonymity attack

Original Data →

#	Non-Sensitive Data			Sensitive Data
	<b>ZIP</b>	<b>Age</b>	<b>Nationality</b>	<b>Condition</b>
1	13053	28	Russian	Heart Disease
2	13068	29	American	Heart Disease
3	13068	21	Japanese	Viral Infection
4	13053	23	American	Viral Infection
5	14853	50	Indian	Cancer
6	14853	55	Russian	Heart Disease
7	14850	47	American	Viral Infection
8	14850	49	American	Viral Infection
9	13053	31	American	Cancer
10	13053	37	Indian	Cancer
11	13068	36	Japanese	Cancer
12	13068	35	American	Cancer

## 4-anonymized Table

Umeko  
Matches  
here

{

Bob  
Matches  
here

{

	<i>Non-Sensitive Data</i>				<i>Sensitive Data</i>
#	<b>ZIP</b>	<b>Age</b>	<b>Nationality</b>	<b>Condition</b>	
1	130**	< 30	*	Heart Disease	
2	130**	< 30	*	Heart Disease	
3	130**	< 30	*	Viral Infection	
4	130**	< 30	*	Viral Infection	
5	1485*	> = 40	*	Cancer	
6	1485*	> = 40	*	Heart Disease	
7	1485*	> = 40	*	Viral Infection	
8	1485*	> = 40	*	Viral Infection	
9	130**	3*	*	Cancer	
10	130**	3*	*	Cancer	
11	130**	3*	*	Cancer	
12	130**	3*	*	Cancer	

# K-Anonymity drawbacks...

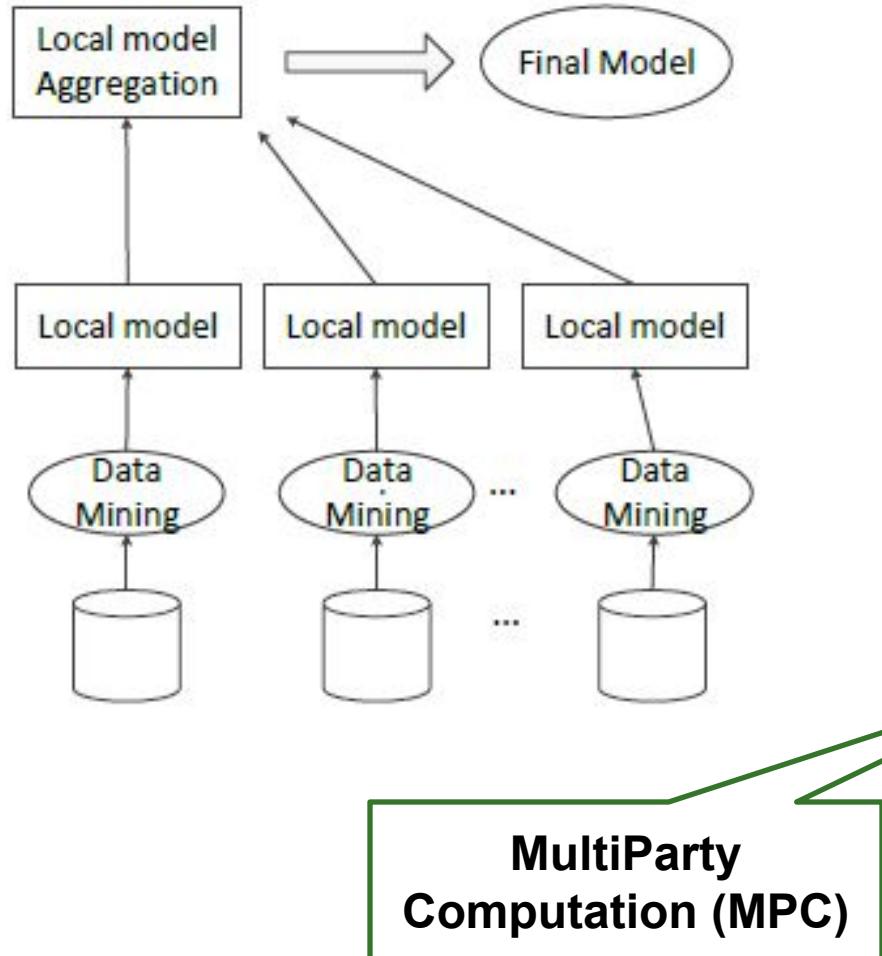
- Basic Reasons for leak –
  - Sensitive attributes lack *diversity* in values
    - Homogeneity Attack
  - Attacker has additional *background knowledge*
    - Background knowledge Attack
- Hence a new solution has been proposed  
*in-addition* to k-anonymity – *l-diversity*

### 3-diverse Table

	<i>Non-Sensitive Data</i>			<i>Sensitive Data</i>
<b>#</b>	<b>ZIP</b>	<b>Age</b>	<b>Nationality</b>	<b>Condition</b>
1	1305*	<= 40	*	Heart Disease
2	1305*	<= 40	*	Viral Infection
3	1305*	<= 40	*	Cancer
4	1305*	<= 40	*	Cancer
5	1485*	>= 40	*	Cancer
6	1485*	>= 40	*	Heart Disease
7	1485*	>= 40	*	Viral Infection
8	1485*	>= 40	*	Viral Infection
9	1306*	<= 40	*	Heart Disease
10	1306*	<= 40	*	Viral Infection
11	1306*	<= 40	*	Cancer
12	1306*	<= 40	*	Cancer

# Cryptography based approaches

# Privacy-Preserving Distributed Data Mining



- Two ways to perform local model aggregation
  - Trusted Third Party(TTP) based approach
  - Collaborative computation based approach

# Privacy-Preserving Distributed Data Mining...

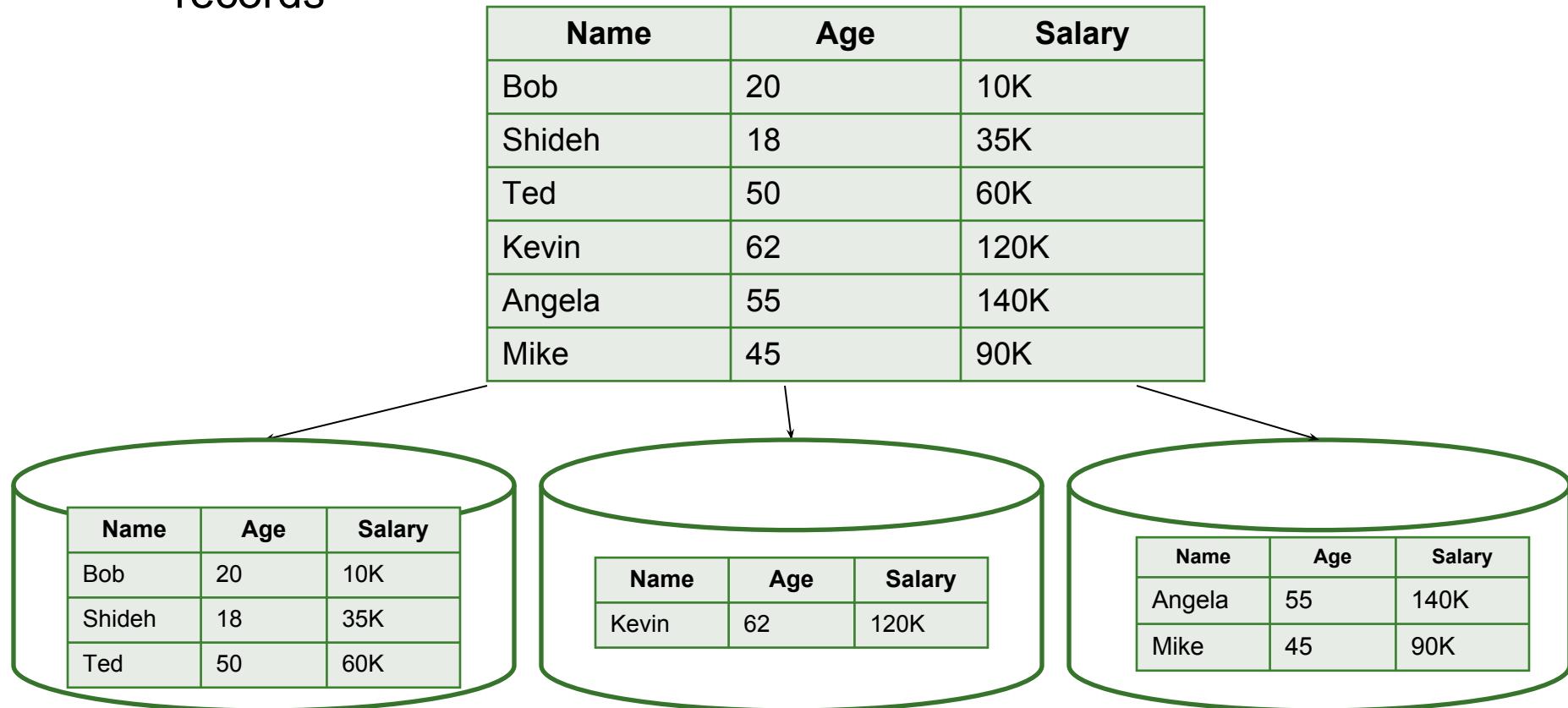
- We focus on the problem of Privacy Preservation in Distributed Data Mining; in particular,
  - The **Secure Multiparty Computation (SMC)**
  - And, the limitations of the approach for solving problems of privacy

# Data Partitioning Models

- Data are partitioned,
  - For better availability, manageability and performance
- Two models
  - Horizontal partitioning
  - Vertical partitioning

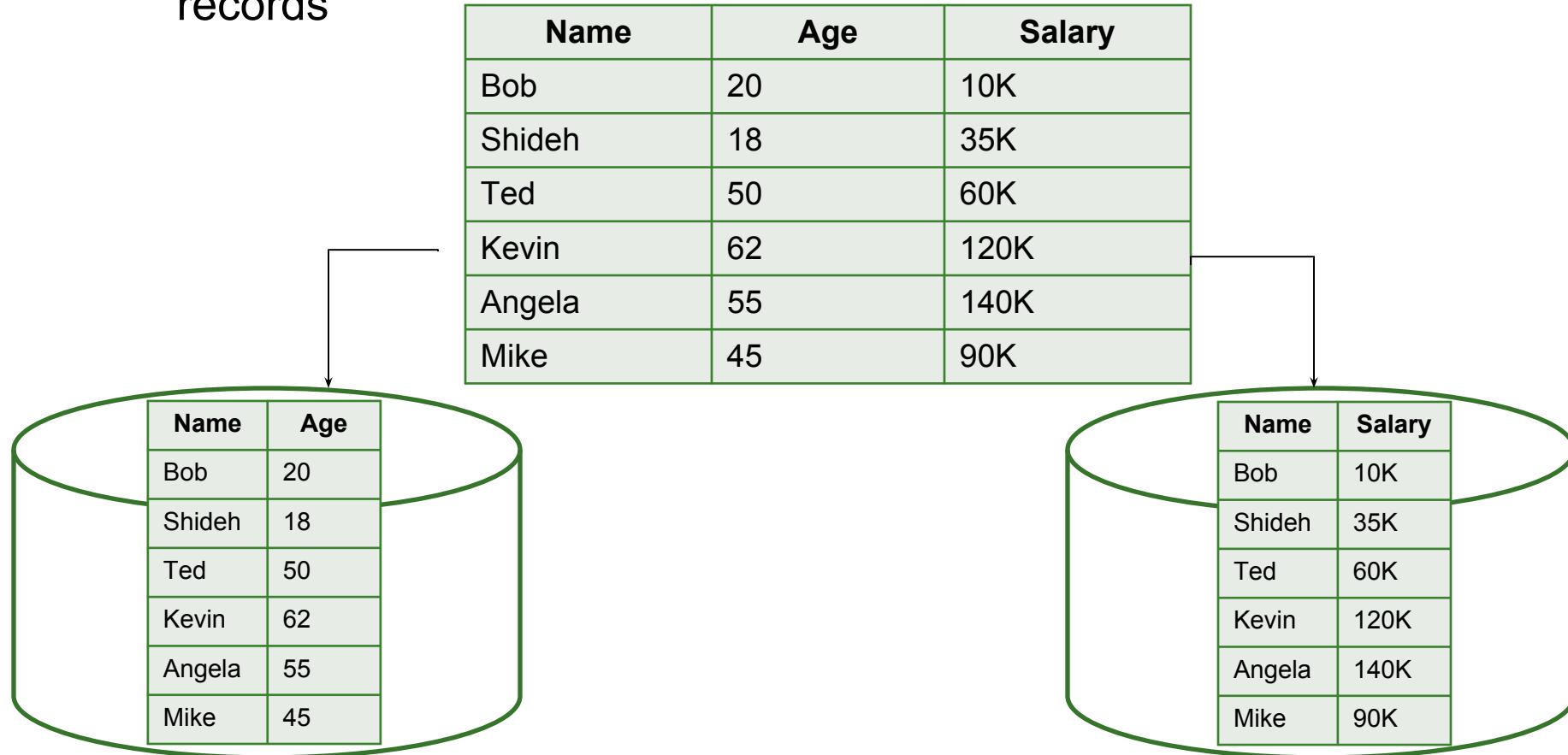
# Data Partitioning Models...

- Horizontal partitioning
  - Different sites collect same information(attributes) about different records



# Data Partitioning Models...

- Vertical partitioning
  - Different sites collect different attributes for the same set of records



# Adversary models

- **Semi-honest**
  - follows the protocol but tries to learn more
- **Malicious**
  - can do anything
  - E.g., Protocol: “Flip a random coin and send the result”
  - Malicious party might...
- Which one is the weaker security model?
- Which one is easy to implement?

# Adversary models...

- Semi-honest model
  - Easier to provide security against semi-honest adversaries
- Malicious model
  - More secure
  - But, is it feasible to change the inputs???
    - Always easy to eavesdrop than to change the inputs !!!

# Basic privacy paradigms

- Secure Computation
  - Oblivious Transfer
    - Random Shares paradigm
  - Homomorphic Encryption
  - Secret Sharing

# Case Studies

- Secure Data Aggregation in Wireless Sensor Networks
- Privacy Preserving Clustering

*Fundamentals of*

# Database Systems

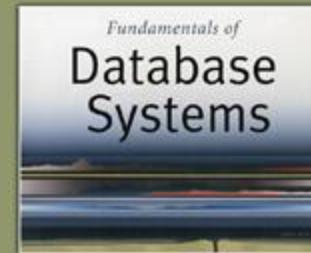


5th Edition

Elmasri / Navathe

# Chapter 23

## Database Security and Authorization



Elmasri / Navathe

# Chapter Outline

- 1 Database Security and Authorization
  - 1.1 Introduction to Database Security Issues
  - 1.2 Types of Security
  - 1.3 Database Security and DBA
  - 1.4 Access Protection, User Accounts, and Database Audits
- 2 Discretionary Access Control Based on Granting Revoking Privileges
  - 2.1 Types of Discretionary Privileges
  - 2.2 Specifying Privileges Using Views
  - 2.3 Revoking Privileges
  - 2.4 Propagation of Privileges Using the GRANT OPTION
  - 2.5 Specifying Limits on Propagation of Privileges

# Chapter Outline (contd.)

- 3 Mandatory Access Control and Role-Based Access Control for Multilevel Security
  - 3.1 Comparing Discretionary Access Control and Mandatory Access Control
  - 3.2 Role-Based Access Control
  - 3.3 Access Control Policies for E-Commerce and the Web
- 4 Introduction to Statistical Database Security
- 5 Introduction to Flow Control
  - 5.1 Covert Channels
- 6 Encryption and Public Key Infrastructures
  - 6.1 The Data and Advanced Encryption Standards
  - 6.2 Public Key Encryption
  - 6.3 Digital Signatures

# 1 Introduction to Database Security Issues

- Types of Security
  - Legal and ethical issues
  - Policy issues
  - System-related issues
  - The need to identify multiple security levels

# Introduction to Database Security Issues (2)

- Threats to databases
  - Loss of **integrity**
  - Loss of **availability**
  - Loss of **confidentiality**
- To protect databases against these types of threats four kinds of countermeasures can be implemented:
  - **Access control**
  - **Inference control**
  - **Flow control**
  - **Encryption**

# Introduction to Database Security Issues (3)

- A DBMS typically includes a database security and authorization subsystem that is responsible for ensuring the security portions of a database against unauthorized access.
- Two types of database security mechanisms:
  - **Discretionary** security mechanisms
  - **Mandatory** security mechanisms

# Introduction to Database Security Issues (4)

- The security mechanism of a DBMS must include provisions for restricting access to the database as a whole
  - This function is called **access control** and is handled by creating user accounts and passwords to control login process by the DBMS.

# Introduction to Database Security Issues (5)

- The security problem associated with databases is that of controlling the access to a **statistical database**, which is used to provide statistical information or summaries of values based on various criteria.
- The countermeasures to **statistical database security** problem is called **inference control measures**.

# Introduction to Database Security Issues (6)

- Another security issue is that of **flow control**, which prevents information from flowing in such a way that it reaches unauthorized users.
- Channels that are pathways for information to flow implicitly in ways that violate the security policy of an organization are called **covert channels**.

# Introduction to Database Security Issues (7)

- A final security issue is **data encryption**, which is used to protect sensitive data (such as credit card numbers) that is being transmitted via some type communication network.
- The data is **encoded** using some **encoding algorithm**.
  - An unauthorized user who access encoded data will have difficulty deciphering it, but authorized users are given decoding or decrypting algorithms (or keys) to decipher data.

## 1.2 Database Security and the DBA

- The database administrator (**DBA**) is the central authority for managing a database system.
  - The DBA's responsibilities include
    - granting privileges to users who need to use the system
    - classifying users and data in accordance with the policy of the organization
- The DBA is responsible for the overall security of the database system.

## 1.2 Database Security and the DBA (2)

- The DBA has a DBA account in the DBMS
  - Sometimes these are called a system or superuser account
  - These accounts provide powerful capabilities such as:
    - 1. Account creation
    - 2. Privilege granting
    - 3. Privilege revocation
    - 4. Security level assignment
  - Action 1 is access control, whereas 2 and 3 are discretionary and 4 is used to control mandatory authorization

# 1.3 Access Protection, User Accounts, and Database Audits

- Whenever a person or group of persons need to access a database system, the individual or group must first apply for a user account.
  - The DBA will then create a new **account id** and **password** for the user if he/she deems there is a legitimate need to access the database
- The user must log in to the DBMS by entering account id and password whenever database access is needed.

# 1.3 Access Protection, User Accounts, and Database Audits(2)

- The database system must also keep **track of all operations** on the database that are applied by a certain user throughout **each login session**.
  - To keep a record of all updates applied to the database and of the particular user who applied each update, we can modify **system log**, which includes an entry for each operation applied to the database that may be required for recovery from a transaction failure or system crash.

# 1.3 Access Protection, User Accounts, and Database Audits(3)

- If any tampering with the database is suspected, a **database audit** is performed
  - A database audit consists of reviewing the log to examine all accesses and operations applied to the database during a certain time period.
- A database log that is used mainly for security purposes is sometimes called an **audit trail**.

# Discretionary Access Control Based on Granting and Revoking Privileges

- The typical method of enforcing **discretionary access control** in a database system is based on the **granting and revoking privileges**.

## 2.1 Types of Discretionary Privileges

- The **account level**:
  - At this level, the DBA specifies the particular privileges that each account holds independently of the relations in the database.
- The **relation level (or table level)**:
  - At this level, the DBA can control the privilege to access each individual relation or view in the database.

## 2.1 Types of Discretionary Privileges(2)

- The privileges at the **account level** apply to the capabilities provided to the account itself and can include
  - the **CREATE SCHEMA** or **CREATE TABLE** privilege, to create a schema or base relation;
  - the **CREATE VIEW** privilege;
  - the **ALTER** privilege, to apply schema changes such adding or removing attributes from relations;
  - the **DROP** privilege, to delete relations or views;
  - the **MODIFY** privilege, to insert, delete, or update tuples;
  - and the **SELECT** privilege, to retrieve information from the database by using a **SELECT** query.

## 2.1 Types of Discretionary Privileges(3)

- The second level of privileges applies to the **relation level**
  - This includes **base relations** and virtual (**view**) relations.
- The granting and revoking of privileges generally follow an authorization model for discretionary privileges known as the access matrix model where
  - The **rows** of a matrix M represents **subjects** (users, accounts, programs)
  - The **columns** represent **objects** (relations, records, columns, views, operations).
  - Each position **M(i,j)** in the matrix represents the types of privileges (read, write, update) that **subject i** holds on **object j**.

## 2.1 Types of Discretionary Privileges(4)

- To control the granting and revoking of relation privileges, each relation R in a database is assigned an **owner account**, which is typically the account that was used when the relation was created in the first place.
  - The owner of a relation is given all privileges on that relation.
  - In SQL2, the DBA can assign an owner to a whole schema by creating the schema and associating the appropriate authorization identifier with that schema, using the **CREATE SCHEMA** command.
  - The owner account holder can **pass privileges** on any of the owned relation to other users by **granting** privileges to their accounts.

## 2.1 Types of Discretionary Privileges(5)

- In SQL the following types of privileges can be granted on each individual relation R:
  - **SELECT** (retrieval or read) privilege on R:
    - Gives the account retrieval privilege.
    - In SQL this gives the account the privilege to use the **SELECT** statement to retrieve tuples from R.
  - **MODIFY** privileges on R:
    - This gives the account the capability to modify tuples of R.
    - In SQL this privilege is further divided into **UPDATE**, **DELETE**, and **INSERT** privileges to apply the corresponding SQL command to R.
    - In addition, both the **INSERT** and **UPDATE** privileges can specify that only certain attributes can be updated by the account.

## 2.1 Types of Discretionary Privileges(6)

- In SQL the following types of privileges can be granted on each individual relation R (contd.):
  - **REFERENCES** privilege on R:
    - This gives the account the capability to **reference** relation R when specifying integrity constraints.
    - The privilege can also be **restricted** to specific attributes of R.
- Notice that to create a **view**, the account must have **SELECT** privilege on all relations involved in the view definition.

## 2.2 Specifying Privileges Using Views

- The mechanism of **views** is an important discretionary authorization mechanism in its own right. For example,
  - If the owner A of a relation R wants another account B to be able to retrieve only some fields of R, then A can create a view V of R that includes only those attributes and then grant SELECT on V to B.
  - The same applies to limiting B to retrieving only certain tuples of R; a view V' can be created by defining the view by means of a query that selects only those tuples from R that A wants to allow B to access.

## 2.3 Revoking Privileges

- In some cases it is desirable to grant a privilege to a user temporarily. For example,
  - The owner of a relation may want to grant the **SELECT** privilege to a user for a specific task and then revoke that privilege once the task is completed.
  - Hence, a mechanism for **revoking** privileges is needed. In SQL, a **REVOKE** command is included for the purpose of **cancelling privileges**.

## 2.4 Propagation of Privileges using the GRANT OPTION

- Whenever the owner A of a relation R grants a privilege on R to another account B, privilege can be given to B with or without the **GRANT OPTION**.
- If the **GRANT OPTION** is given, this means that B can also grant that privilege on R to other accounts.
  - Suppose that B is given the **GRANT OPTION** by A and that B then grants the privilege on R to a third account C, also with **GRANT OPTION**. In this way, privileges on R can **propagate** to other accounts without the knowledge of the owner of R.
  - If the owner account A now revokes the privilege granted to B, all the privileges that B propagated based on that privilege should automatically be revoked by the system.

## 2.5 An Example

- Suppose that the DBA creates four accounts
  - A1, A2, A3, A4
- and wants only A1 to be able to create base relations.

Then the DBA must issue the following GRANT command in SQL

**GRANT** CREATETAB TO A1;

- In SQL2 the same effect can be accomplished by having the DBA issue a **CREATE SCHEMA** command as follows:

**CREATE SCHEMA EXAMPLE AUTHORIZATION**  
A1;

## 2.5 An Example(2)

- User account **A1** can create tables under the schema called **EXAMPLE**.
- Suppose that A1 **creates** the two base relations **EMPLOYEE** and **DEPARTMENT**
  - A1 is then **owner** of these two relations and hence all the relation privileges on each of them.
- Suppose that A1 wants to grant A2 the privilege to insert and delete tuples in both of these relations, but A1 does not want A2 to be able to propagate these privileges to additional accounts:

**GRANT INSERT, DELETE ON  
EMPLOYEE, DEPARTMENT TO A2 ;**

## 2.5 An Example(3)

**EMPLOYEE**

Name	Ssn	Bdate	Address	Sex	Salary	Dno
------	-----	-------	---------	-----	--------	-----

**DEPARTMENT**

Dnumber	Dname	Mgr_ssn
---------	-------	---------

**Figure 23.1**

Schemas for the two relations EMPLOYEE and DEPARTMENT.

## 2.5 An Example(4)

- Suppose that A1 wants to allow A3 to retrieve information from either of the two tables and also to be able to propagate the SELECT privilege to other accounts.
- A1 can issue the command:

**GRANT SELECT ON EMPLOYEE, DEPARTMENT  
TO A3 WITH GRANT OPTION;**

- A3 can grant the **SELECT** privilege on the **EMPLOYEE** relation to A4 by issuing:

**GRANT SELECT ON EMPLOYEE TO A4;**

- Notice that A4 can't propagate the SELECT privilege because GRANT OPTION was not given to A4

## 2.5 An Example(5)

- Suppose that A1 decides to revoke the SELECT privilege on the EMPLOYEE relation from A3; A1 can issue:

**REVOKE SELECT ON EMPLOYEE FROM A3;**

- The DBMS must now automatically revoke the SELECT privilege on EMPLOYEE from A4, too, because A3 granted that privilege to A4 and A3 does not have the privilege any more.

## 2.5 An Example(6)

- Suppose that A1 wants to give back to A3 a limited capability to **SELECT** from the **EMPLOYEE** relation and wants to allow A3 to be able to propagate the privilege.

- The limitation is to retrieve only the **NAME**, **BDATE**, and **ADDRESS** attributes and only for the tuples with **DNO=5**.

- A1 then create the view:

```
CREATE VIEW A3EMPLOYEE AS
    SELECT NAME, BDATE, ADDRESS
    FROM EMPLOYEE
    WHERE DNO = 5;
```

- After the view is created, A1 can grant **SELECT** on the view **A3EMPLOYEE** to A3 as follows:

```
GRANT SELECT ON A3EMPLOYEE TO A3
WITH GRANT OPTION;
```

## 2.5 An Example(7)

- Finally, suppose that A1 wants to allow A4 to update only the SALARY attribute of EMPLOYEE;
- A1 can issue:

```
GRANT UPDATE ON EMPLOYEE (SALARY) TO  
A4 ;
```

- The **UPDATE** or **INSERT** privilege can specify particular attributes that may be updated or inserted in a relation.
- Other privileges (**SELECT**, **DELETE**) are not attribute specific.

## 2.6 Specifying Limits on Propagation of Privileges

- Techniques to limit the propagation of privileges have been developed, although they have not yet been implemented in most DBMSs and are not a part of SQL.
  - Limiting **horizontal propagation** to an integer number  $i$  means that an account  $B$  given the GRANT OPTION can grant the privilege to at most  $i$  other accounts.
  - **Vertical propagation** is more complicated; it limits the depth of the granting of privileges.

# 3 Mandatory Access Control and Role-Based Access Control for Multilevel Security

- The discretionary access control techniques of granting and revoking privileges on relations has traditionally been the main security mechanism for relational database systems.
- This is an all-or-nothing method:
  - A user either has or does not have a certain privilege.
- In many applications, and **additional security policy** is needed that classifies data and users based on security classes.
  - This approach as **mandatory access control**, would typically be **combined** with the discretionary access control mechanisms.

## 3 Mandatory Access Control and Role-Based Access Control for Multilevel Security(2)

- Typical **security classes** are top secret (TS), secret (S), confidential (C), and unclassified (U), where TS is the highest level and U the lowest:  $TS \geq S \geq C \geq U$
- The commonly used model for multilevel security, known as the Bell-LaPadula model, classifies each **subject** (user, account, program) and **object** (relation, tuple, column, view, operation) into one of the security classifications, T, S, C, or U:
  - **Clearance** (classification) of a subject S as **class(S)** and to the **classification** of an object O as **class(O)**.

## 3 Mandatory Access Control and Role-Based Access Control for Multilevel Security(3)

- Two restrictions are enforced on data access based on the subject/object classifications:
  - **Simple security property:** A subject S is not allowed read access to an object O unless  $\text{class}(S) \geq \text{class}(O)$ .
  - A subject S is not allowed to write an object O unless  $\text{class}(S) \leq \text{class}(O)$ . This known as the **star property** (or \* property).

### 3 Mandatory Access Control and Role-Based Access Control for Multilevel Security(4)

- To incorporate multilevel security notions into the relational database model, it is common to consider attribute values and tuples as data objects.
- Hence, each attribute A is associated with a **classification attribute C** in the schema, and each attribute value in a tuple is associated with a corresponding security classification.
- In addition, in some models, a **tuple classification** attribute TC is added to the relation attributes to provide a classification for each tuple as a whole.
- Hence, a **multilevel relation** schema R with n attributes would be represented as
  - $R(A_1, C_1, A_2, C_2, \dots, A_n, C_n, TC)$
- where each  $C_i$  represents the classification attribute associated with attribute  $A_i$ .

## 3 Mandatory Access Control and Role-Based Access Control for Multilevel Security(5)

- The value of the **TC** attribute in each tuple  $t$  – which is the highest of all attribute classification values within  $t$  – provides a general classification for the tuple itself, whereas each  $C_i$  provides a finer security classification for each attribute value within the tuple.
  - The apparent key of a multilevel relation is the set of attributes that would have formed the primary key in a regular(single-level) relation.

## 3 Mandatory Access Control and Role-Based Access Control for Multilevel Security(6)

- A multilevel relation will appear to contain different data to subjects (users) with different clearance levels.
  - In some cases, it is possible to store a single tuple in the relation at a higher classification level and produce the corresponding tuples at a lower-level classification through a process known as **filtering**.
  - In other cases, it is necessary to store two or more tuples at different classification levels with the same value for the **apparent key**.
- This leads to the concept of **polyinstantiation** where several tuples can have the same apparent key value but have different attribute values for users at different classification levels.

## 3 Mandatory Access Control and Role-Based Access Control for Multilevel Security(7)

- In general, the **entity integrity** rule for multilevel relations states that all attributes that are members of the apparent key must not be null and must have the same security classification within each individual tuple.
- In addition, all other attribute values in the tuple must have a security classification greater than or equal to that of the apparent key.
  - This **constraint** ensures that a user can see the key if the user is permitted to see any part of the tuple at all.

## 3 Mandatory Access Control and Role-Based Access Control for Multilevel Security(8)

- Other integrity rules, called **null integrity** and **interinstance integrity**, informally ensure that if a tuple value at some security level can be filtered (derived) from a higher-classified tuple, then it is sufficient to store the higher-classified tuple in the multilevel relation.

## 3.1 Comparing Discretionary Access Control and Mandatory Access Control

- **Discretionary Access Control (DAC)** policies are characterized by a high degree of flexibility, which makes them suitable for a large variety of application domains.
  - The main drawback of **DAC** models is their vulnerability to malicious attacks, such as Trojan horses embedded in application programs.

## 3.1 Comparing Discretionary Access Control and Mandatory Access Control(2)

- By contrast, mandatory policies ensure a high degree of protection in a way, they prevent any illegal flow of information.
- Mandatory policies have the drawback of being too rigid and they are only applicable in limited environments.
- In many practical situations, discretionary policies are preferred because they offer a better trade-off between security and applicability.

## 3.2 Role-Based Access Control

- **Role-based access control (RBAC)** emerged rapidly in the 1990s as a proven technology for managing and enforcing security in large-scale enterprise-wide systems.
- Its basic notion is that permissions are associated with roles, and users are assigned to appropriate roles.
- Roles can be created using the **CREATE ROLE** and **DESTROY ROLE** commands.
  - The **GRANT** and **REVOKE** commands discussed under DAC can then be used to assign and revoke privileges from roles.

## 3.2 Role-Based Access Control(2)

- **RBAC** appears to be a viable alternative to traditional discretionary and mandatory access controls; it ensures that only authorized users are given access to certain data or resources.
- Many DBMSs have allowed the concept of roles, where privileges can be assigned to roles.
- Role hierarchy in **RBAC** is a natural way of organizing roles to reflect the organization's lines of authority and responsibility.

## 3.2 Role-Based Access Control(3)

- Another important consideration in **RBAC** systems is the possible temporal constraints that may exist on roles, such as time and duration of role activations, and timed triggering of a role by an activation of another role.
- Using an **RBAC** model is highly desirable goal for addressing the key security requirements of Web-based applications.
- In contrast, discretionary access control (**DAC**) and mandatory access control (**MAC**) models **lack capabilities** needed to support the security requirements emerging enterprises and Web-based applications.

### 3.3 Access Control Policies for E-Commerce and the Web

- **E-Commerce environments** require elaborate policies that go beyond traditional DBMSs.
  - In an e-commerce environment the resources to be protected are not only traditional data but also knowledge and experience.
  - The access control mechanism should be flexible enough to support a wide spectrum of heterogeneous protection objects.
- A related requirement is the support for **content-based access-control**.

### 3.3 Access Control Policies for E-Commerce and the Web(2)

- Another requirement is related to the heterogeneity of subjects, which requires access control policies based on user characteristics and qualifications.
  - A possible solution, to better take into account user profiles in the formulation of access control policies, is to support the notion of credentials.
  - A **credential** is a set of properties concerning a user that are relevant for security purposes
    - For example, age, position within an organization
  - It is believed that the XML language can play a key role in access control for e-commerce applications.

# 4 Introduction to Statistical Database Security

- **Statistical databases** are used mainly to produce statistics on various populations.
- The database may contain **confidential data** on individuals, which should be protected from user access.
- Users are permitted to retrieve **statistical information** on the populations, such as **averages, sums, counts, maximums, minimums, and standard deviations**.

## 4 Introduction to Statistical Database Security(2)

- A **population** is a set of tuples of a relation (table) that satisfy some selection condition.
- Statistical queries involve applying **statistical functions** to a **population** of tuples.

# 4 Introduction to Statistical Database Security(3)

- For example, we may want to retrieve the *number* of individuals in a **population** or the *average income* in the population.
  - However, statistical users are not allowed to retrieve individual data, such as the income of a specific person.
- Statistical database security techniques must prohibit the retrieval of individual data.
- This can be achieved by prohibiting queries that retrieve attribute values and by allowing only queries that involve statistical aggregate functions such as COUNT, SUM, MIN, MAX, AVERAGE, and STANDARD DEVIATION.
  - Such queries are sometimes called **statistical queries**.

# 4 Introduction to Statistical Database Security(4)

- It is DBMS's responsibility to ensure confidentiality of information about individuals, while still providing useful statistical summaries of data about those individuals to users. Provision of **privacy protection** of users in a statistical database is paramount.
- In some cases it is possible to **infer** the values of individual tuples from a sequence statistical queries.
  - This is particularly true when the conditions result in a population consisting of a small number of tuples.

# 5 Introduction to Flow Control

- **Flow control** regulates the distribution or flow of information among accessible objects.
- A **flow** between object X and object Y occurs when a program reads values from X and writes values into Y.
  - Flow controls check that information contained in some objects does not flow explicitly or implicitly into less protected objects.
- A **flow policy** specifies the channels along which information is allowed to move.
  - The simplest flow policy specifies just two classes of information:
    - confidential (C) and nonconfidential (N)
    - and allows all flows except those from class C to class N.

## 5.1 Covert Channels

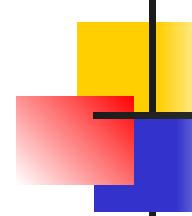
- A **covert channel** allows a transfer of information that violates the security or the policy.
- A **covert channel allows** information to pass from a higher classification level to a lower classification level through **improper means**.

## 5.1 Covert Channels(2)

- **Covert channels** can be classified into two broad categories:
  - **Storage channels** do not require any temporal synchronization, in that information is conveyed by accessing system information or what is otherwise inaccessible to the user.
  - **Timing channel** allow the information to be conveyed by the timing of events or processes.
- Some security experts believe that one way to avoid covert channels is for programmers to not actually gain access to sensitive data that a program is supposed to process after the program has been put into operation.

## Chapter 16

### *Security at the Application Layer: PGP and S/MIME*



## Chapter 16

### Objectives

- To explain the general structure of an e-mail application program
- To discuss how PGP can provide security services for e-mail
- To discuss how S/MIME can provide security services for e-mail
- To define trust mechanism in both PGP and S/MIME
- To show the structure of messages exchanged in PGP and S/MIME

# 16-1 E-MAIL

*Let us first discuss the electronic mail (e-mail) system in general.*

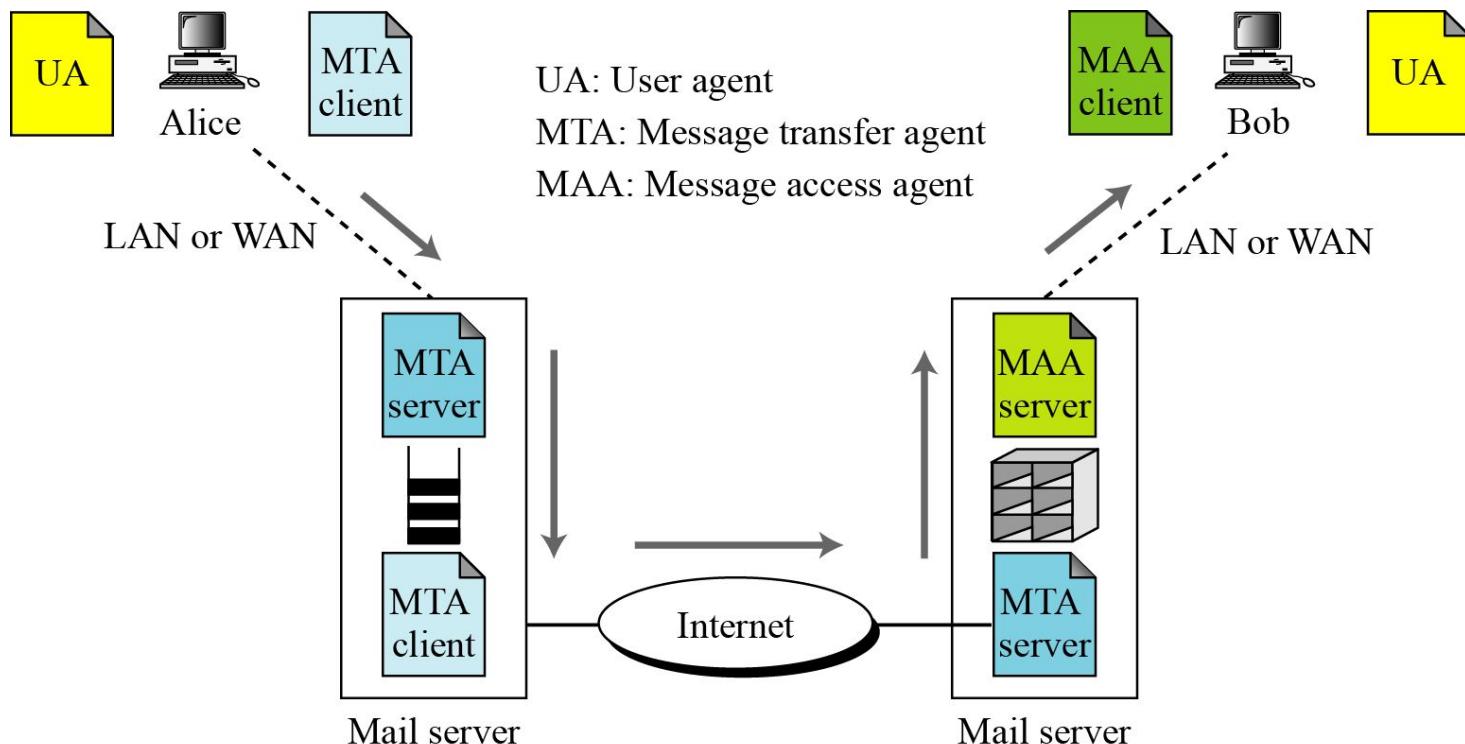
**Topics discussed in this section:**

**16.1.1 E-mail Architecture**

**16.1.2 E-mail Security**

## 16.1.1 E-mail Architecture

Figure 16.1 E-mail architecture



## *16.1.2 E-mail Security*

### *Cryptographic Algorithms*

#### **Note**

**In e-mail security, the sender of the message needs to include the name or identifiers of the algorithms used in the message.**

### *Certificates*

*It is obvious that some public-key algorithms must be used for e-mail security.*

## *16.1.2 Continued*

### *Cryptographic Secrets*

#### **Note**

**In e-mail security, the encryption/decryption is done using a symmetric-key algorithm, but the secret key to decrypt the message is encrypted with the public key of the receiver and is sent with the message.**

## 16-2 PGP

*Pretty Good Privacy (PGP) can be used to create a secure e-mail message or to store a file securely for future retrieval.*

### **Topics discussed in this section:**

**16.2.1 Scenarios**

**16.2.2 Key Rings**

**16.2.3 PGP Certificates**

**16.2.4 Key Revocation**

**16.2.5 Extracting Information from Rings**

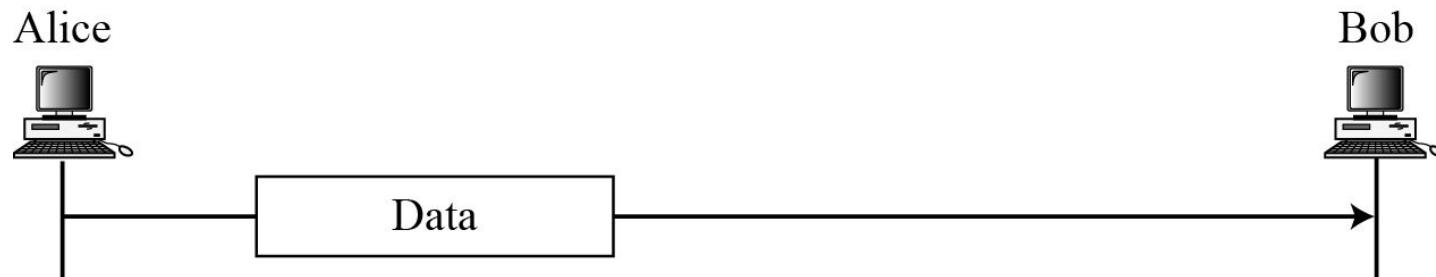
**16.2.6 PGP Packets**

**16.2.7 PGP Messages**

## 16.2.1 Scenarios

### Plaintext

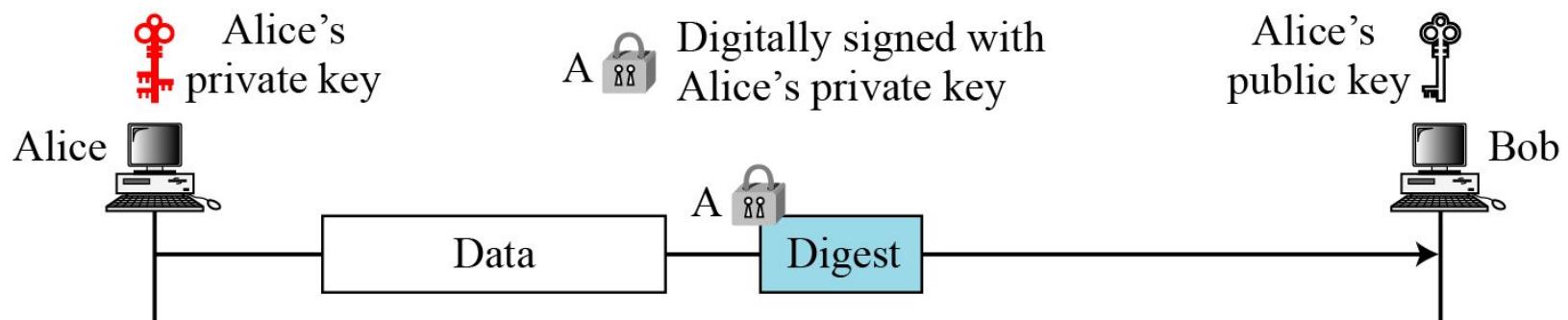
**Figure 16.2 A plaintext message**



## 16.2.1 *Continued*

### *Message Integrity*

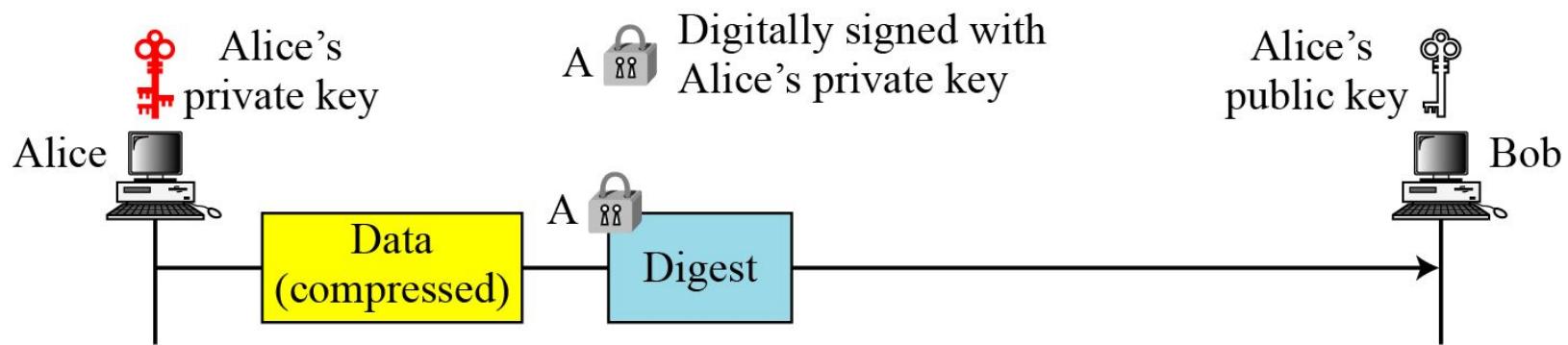
**Figure 16.3** *An authenticated message*



## 16.2.1 *Continued*

### Compression

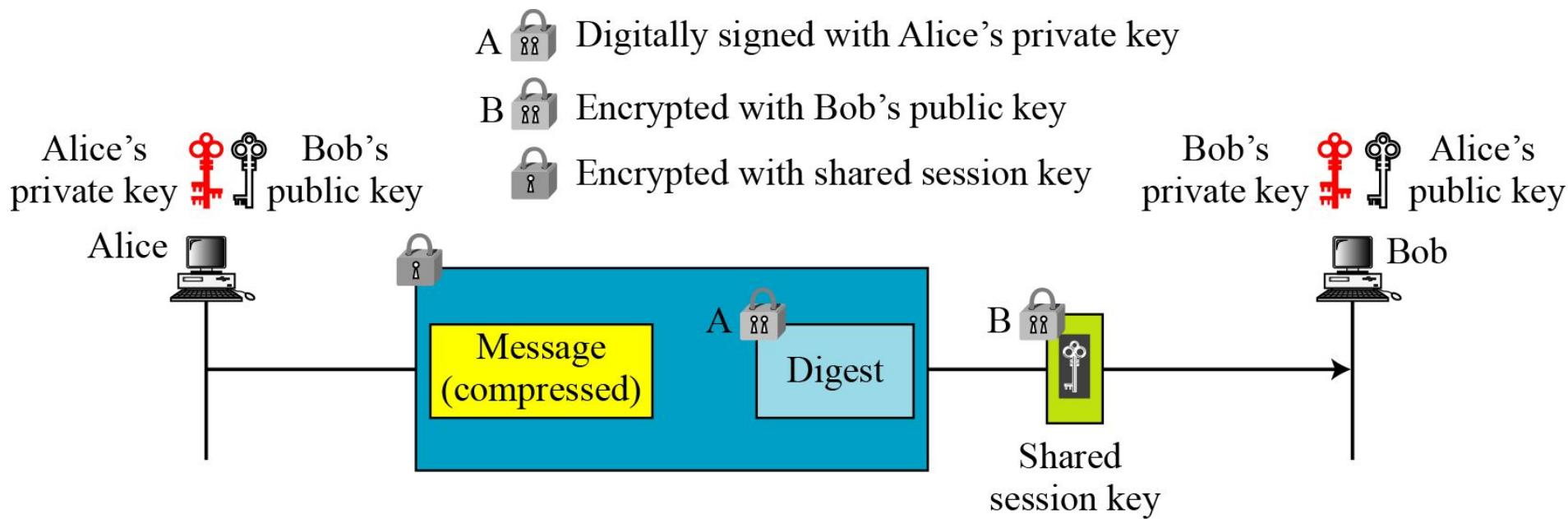
**Figure 16.4** *A compressed message*

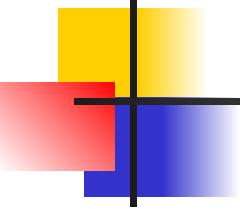


## 16.2.1 Continued

### Confidentiality with One-Time Session Key

Figure 16.5 A confidential message





## *16.2.1 Continued*

### *Code Conversion*

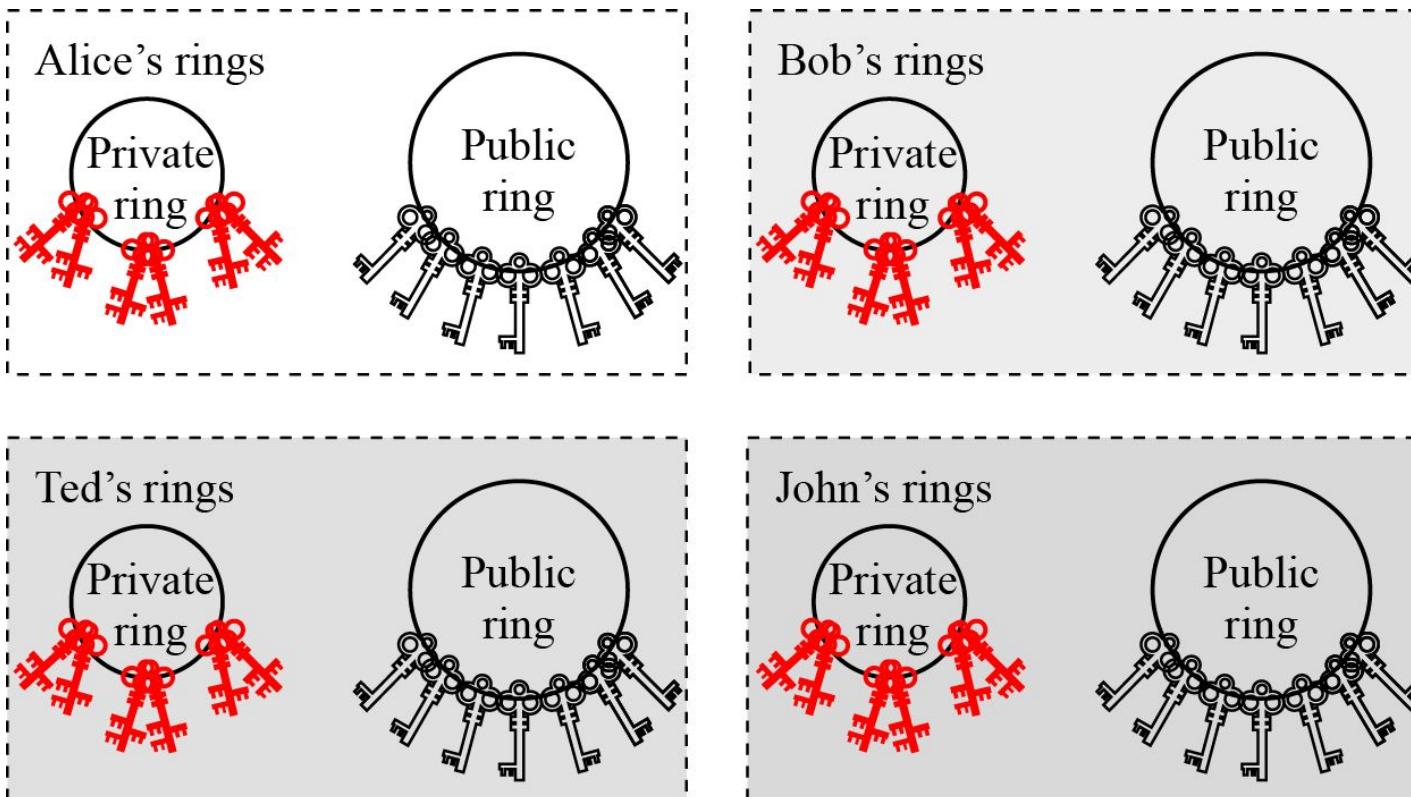
*Another service provided by PGP is code conversion.  
PGP uses Radix-64 conversion.*

### *Segmentation*

*PGP allows segmentation of the message.*

## 16.2.2 Key Rings

Figure 16.6 Key rings in PGP



## 16.2.2 *Continued*

### *PGP Algorithms*

**Table 16.1** *Public-key algorithms*

<i>ID</i>	<i>Description</i>
1	RSA (encryption or signing)
2	RSA (for encryption only)
3	RSA (for signing only)
16	ElGamal (encryption only)
17	DSS
18	Reserved for elliptic curve
19	Reserved for ECDSA
20	ElGamal (for encryption or signing)
21	Reserved for Diffie-Hellman
100–110	Private algorithms

## 16.2.2 *Continued*

**Table 16.2** *Symmetric-key algorithms*

<i>ID</i>	<i>Description</i>
0	No Encryption
1	IDEA
2	Triple DES
3	CAST-128
4	Blowfish
5	SAFER-SK128
6	Reserved for DES/SK
7	Reserved for AES-128
8	Reserved for AES-192
9	Reserved for AES-256
100–110	Private algorithms

## 16.2.2 *Continued*

**Table 16.3** *Hash Algorithms*

<i>ID</i>	<i>Description</i>
1	MD5
2	SHA-1
3	RIPE-MD/160
4	Reserved for double-width SHA
5	MD2
6	TIGER/192
7	Reserved for HAVAL
100–110	Private algorithms

## 16.2.2 *Continued*

**Table 16.4** *Compression methods*

<i>ID</i>	<i>Description</i>
0	Uncompressed
1	ZIP
2	ZLIP
100–110	Private methods

## *16.2.3 PGP Certificates*

### *X.509 Certificates*

*Protocols that use X.509 certificates depend on the hierarchical structure of the trust.*

#### **Note**

**In X.509, there is a single path from the fully trusted authority to any certificate.**

## 16.2.3 Continued

### *PGP Certificates*

*In PGP, there is no need for CAs; anyone in the ring can sign a certificate for anyone else in the ring.*

#### **Note**

In PGP, there can be multiple paths from fully or partially trusted authorities to any subject.

### *Trusts and Legitimacy*

*The entire operation of PGP is based on introducer trust, the certificate trust, and the legitimacy of the public keys.*

## 16.2.3 *Continued*

**Figure 16.7** *Format of private key ring table*



User ID	Key ID	Public key	Encrypted private key	Timestamp
⋮	⋮	⋮	⋮	⋮

## 16.2.3 *Continued*

### Example 16.1

Let us show a private key ring table for Alice. We assume that Alice has only two user IDs, **alice@some.com** and **alice@anet.net**. We also assume that Alice has two sets of private/public keys, one for each user ID.

**Table 16.5** *Private key ring table for Example 1*

User ID	Key ID	Public Key	Encrypted Private Key	Timestamp
alice@anet.net	AB13...45	AB13...45...59	<b>32452398...23</b>	031505-16:23
alice@some.com	FA23...12	FA23...12...22	<b>564A4923...23</b>	031504-08:11

## 16.2.3 *Continued*

**Figure 16.8** *Format of a public key ring table*



User ID	Key ID	Public key	Producer trust	Certificate(s)	Certificate trust(s)	Key Legitimacy	Timestamp
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

## 16.2.3 *Continued*

### Example 16.2

A series of steps will show how a public key ring table is formed for Alice.

**Table 16.6** *Example 2, starting table*

User ID	Key ID	Public key	Prod. trust	Certificate	Cert. trust	Key legit.	Time-stamp
Alice...	AB...	AB.....	F			F	.....

**Table 16.7** *Example 2, after Bob is added to the table*

User ID	Key ID	Public key	Prod. trust	Certificate	Cert. trust	Key legit.	Time-stamp
Alice...	AB...	AB.....	F			F	.....
Bob...	12...	12.....	F			F	.....

## 16.2.3 *Continued*

### Example 16.2 *Continued*

**Table 16.8** Example 2, after Ted is added to the table

User ID	Key ID	Public key	Prod. trust	Certificate	Cert. trust	Key legit.	Time-stamp
Alice...	AB...	AB.....	F			F	.....
Bob...	12...	12.....	F			F	.....
Ted...	48...	48.....	F	Bob's	F	F	.....

**Table 16.9** Example 2, after Anne is added to the table

User ID	Key ID	Public key	Prod. trust	Certificate	Cert. trust	Key legit.	Time-stamp
Alice...	AB...	AB.....	F			F	.....
Bob...	12...	12.....	F			F	.....
Ted...	48...	48.....	F	Bob's	F	F	.....
Anne...	71...	71.....	P	Bob's	F	F	.....

## 16.2.3 *Continued*

### Example 16.2 *Continued*

**Table 16.10** Example 2, after John is added to the table

User ID	Key ID	Public key	Prod. Trust	Certificate	Cert. trust	Key legit.	Time-stamp
Alice...	AB...	AB.....	F			F	.....
Bob...	12...	12.....	F			F	.....
Ted...	48...	48.....	F	Bob's	F	F	.....
Anne...	71...	71.....	P	Bob's	F	F	.....
John...	31...	31.....	N	Anne's	P	P	.....

## 16.2.3 *Continued*

### Example 16.2 *Continued*

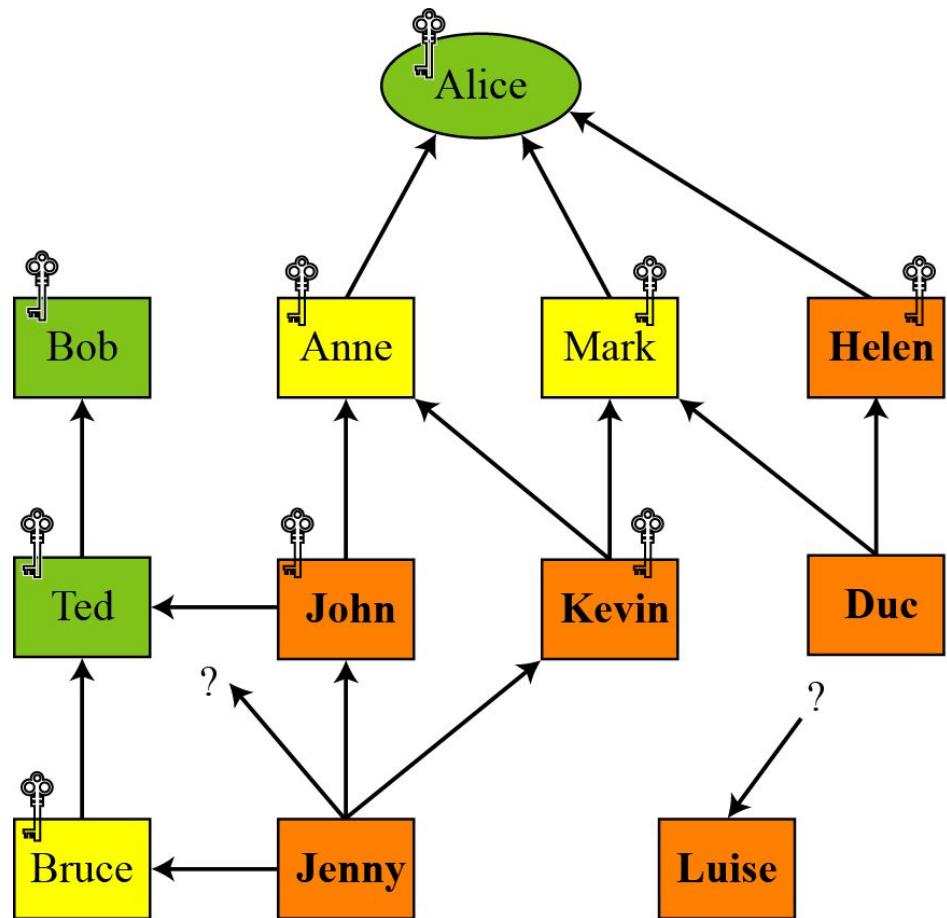
**Table 16.11** *Example 2, after one more certificate received for John*

User ID	Key ID	Public key	Prod. trust	Certificate	Cert. trust	Key legit.	Time-stamp
Alice...	AB...	AB.....	F			F	.....
Bob...	12...	12.....	F			F	.....
Ted...	48...	48.....	F	Bob's	F	F	.....
Anne...	71...	71.....	P	Bob's	F	F	.....
John...	31...	31.....	N	Anne's Ted's	P F	F	.....

## 16.2.3 Continued

### Trust Model in PGP

Figure 16.9 Trust model

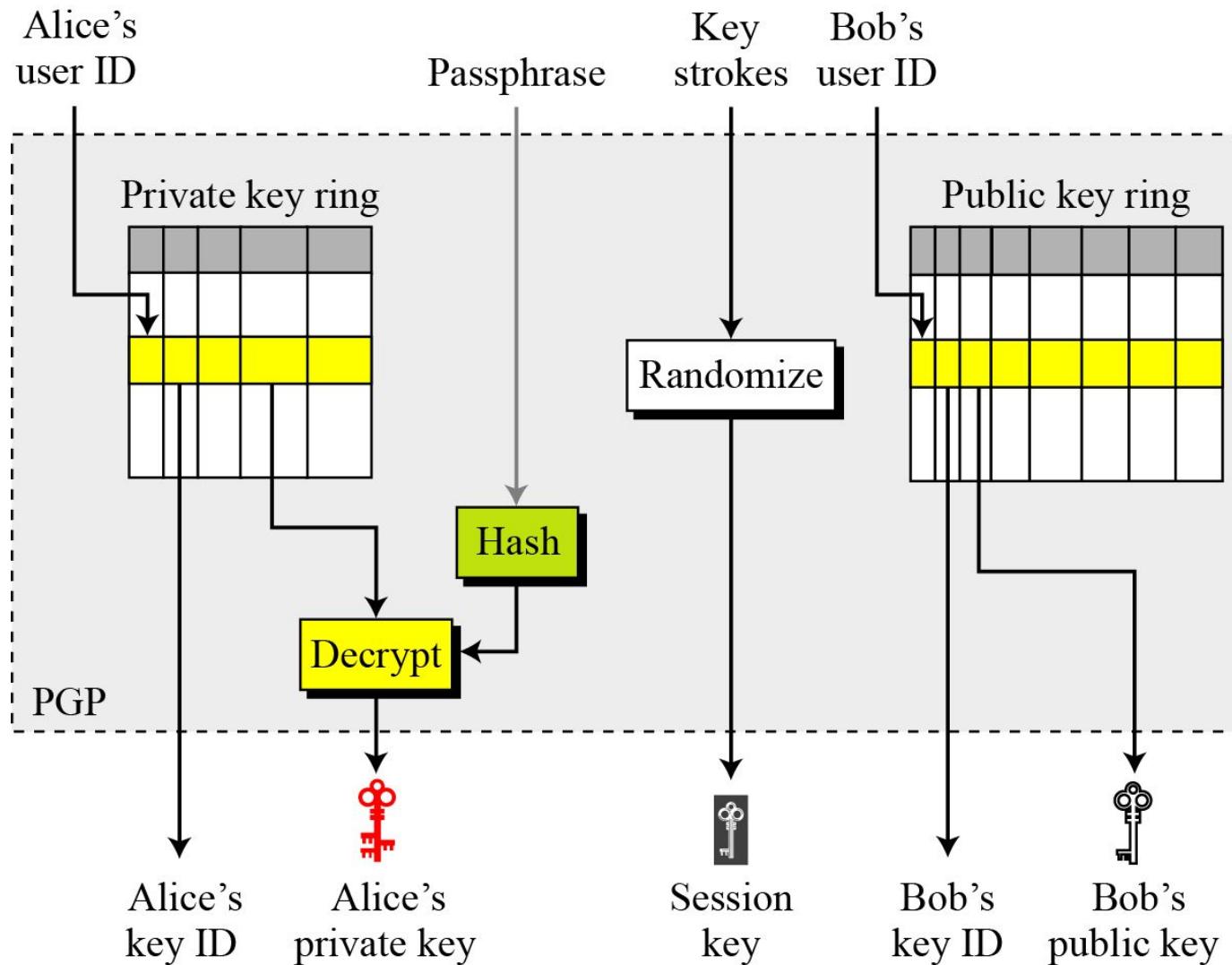


## *16.2.4 Key Revocation*

*It may become necessary for an entity to revoke his or her public key from the ring. This may happen if the owner of the key feels that the key is compromised (stolen, for example) or just too old to be safe.*

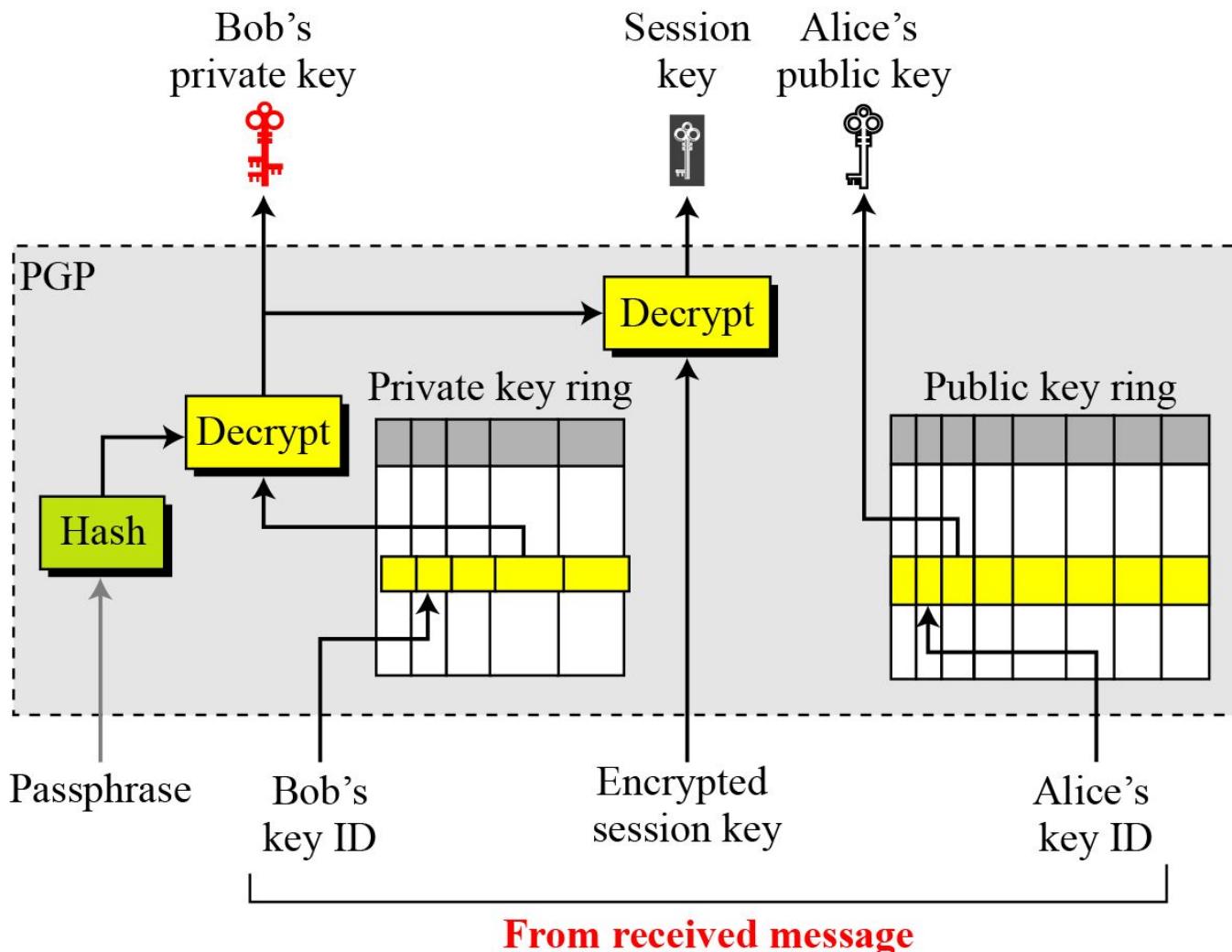
## 16.2.5 Extracting Information from Rings

Figure 16.10 Extracting information at the sender site



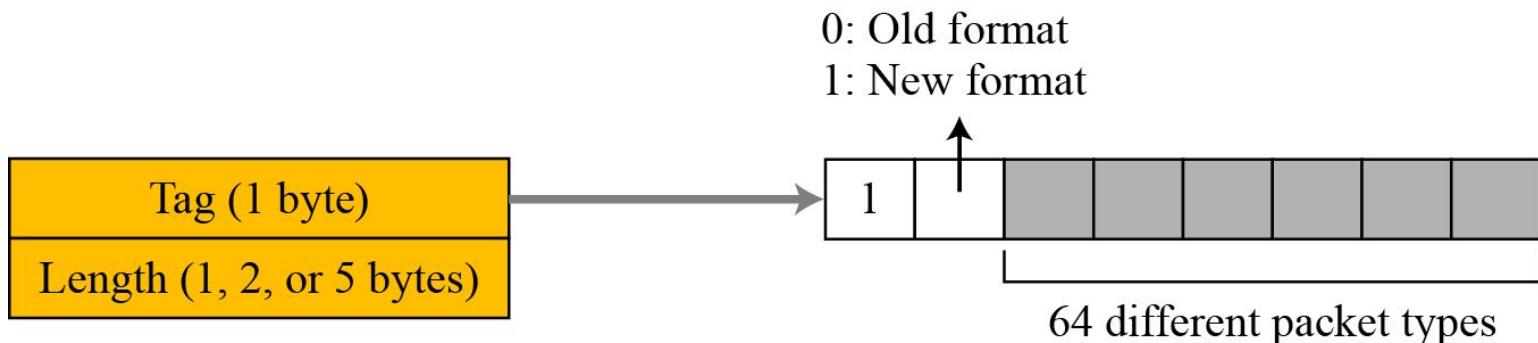
## 16.2.5 Continued

**Figure 16.11** Extracting information at the receiver site



## 16.2.6 PGP Packets

**Figure 16.12** Format of packet header



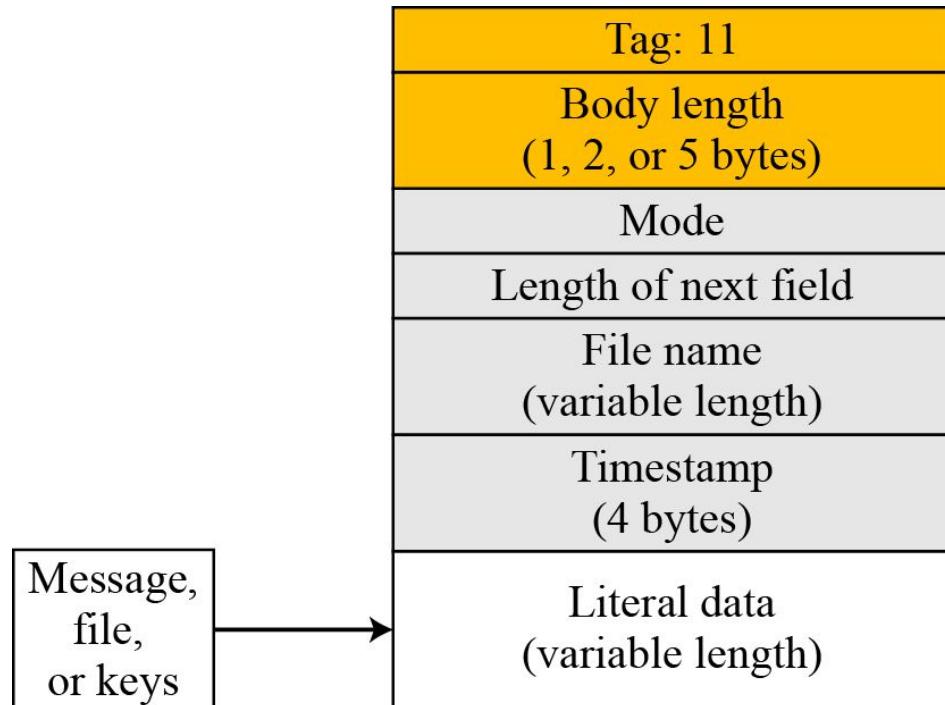
## 16.2.6 *Continued*

**Table 16.12** *Some commonly used packet types*

<i>Value</i>	<i>Packet type</i>
1	Session key packet encrypted using a public key
2	Signature packet
5	Private-key packet
6	Public-key packet
8	Compressed data packet
9	Data packet encrypted with a secret key
11	Literal data packet
13	User ID packet

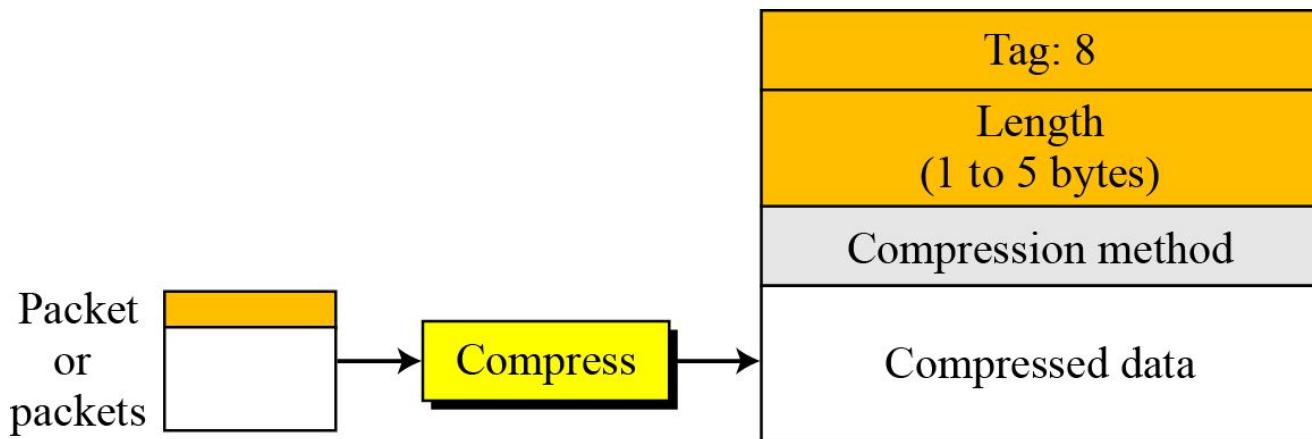
## 16.2.6 *Continued*

**Figure 16.13** *Literal data packet*



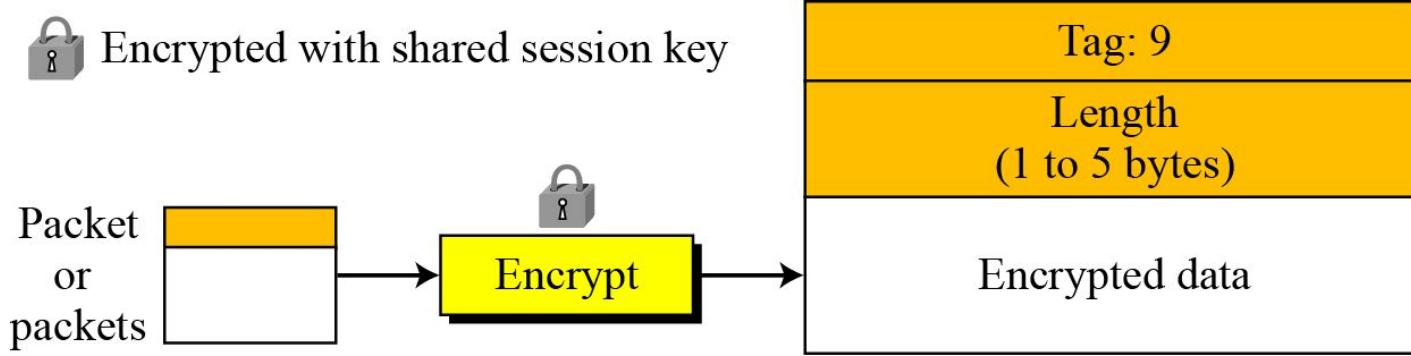
## 16.2.6 *Continued*

**Figure 16.14** Compressed data packet



## 16.2.6 *Continued*

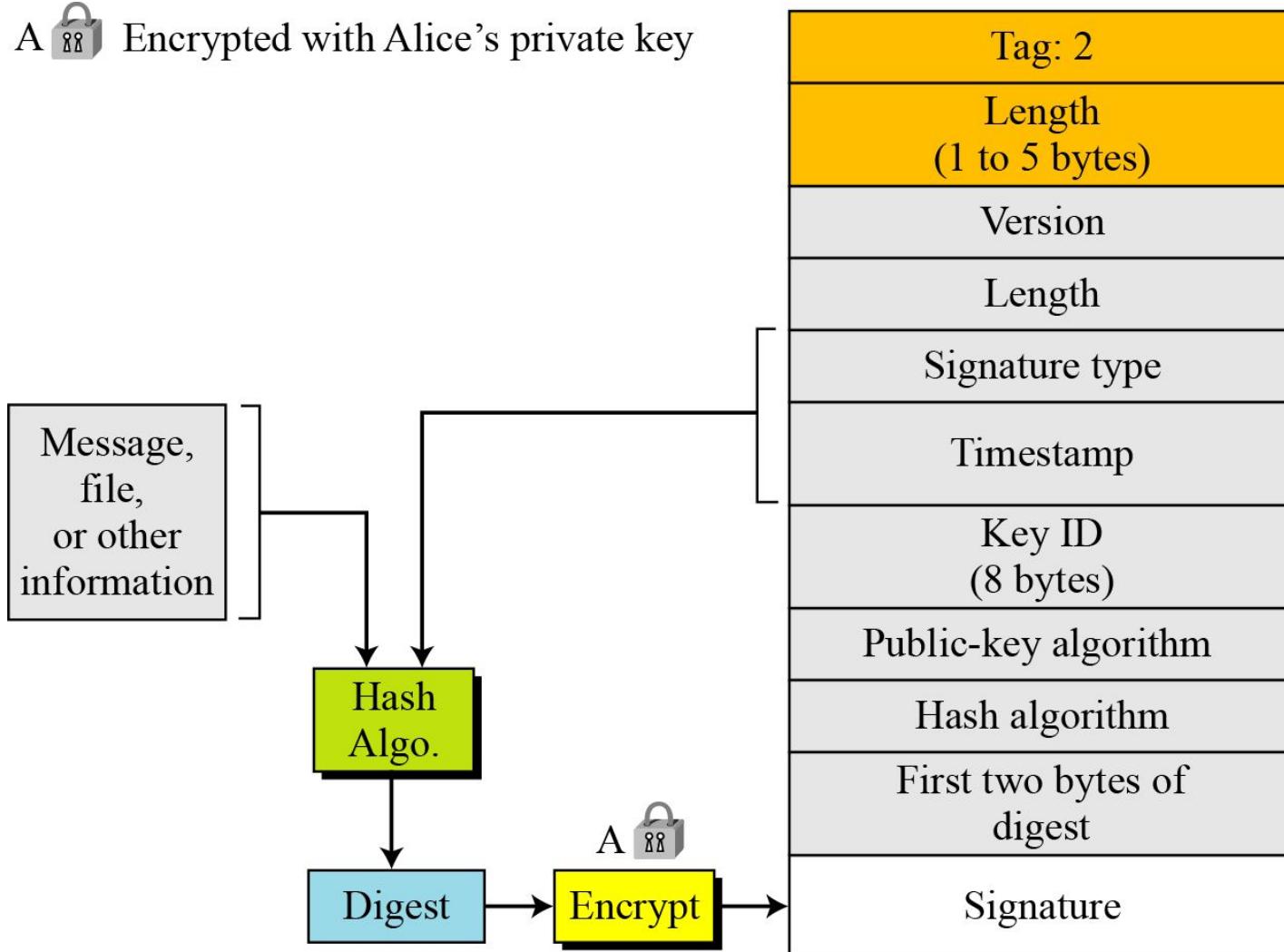
**Figure 16.15** *Encrypted data packet*



## 16.2.6 Continued

Figure 16.16 Signature packet

A Encrypted with Alice's private key



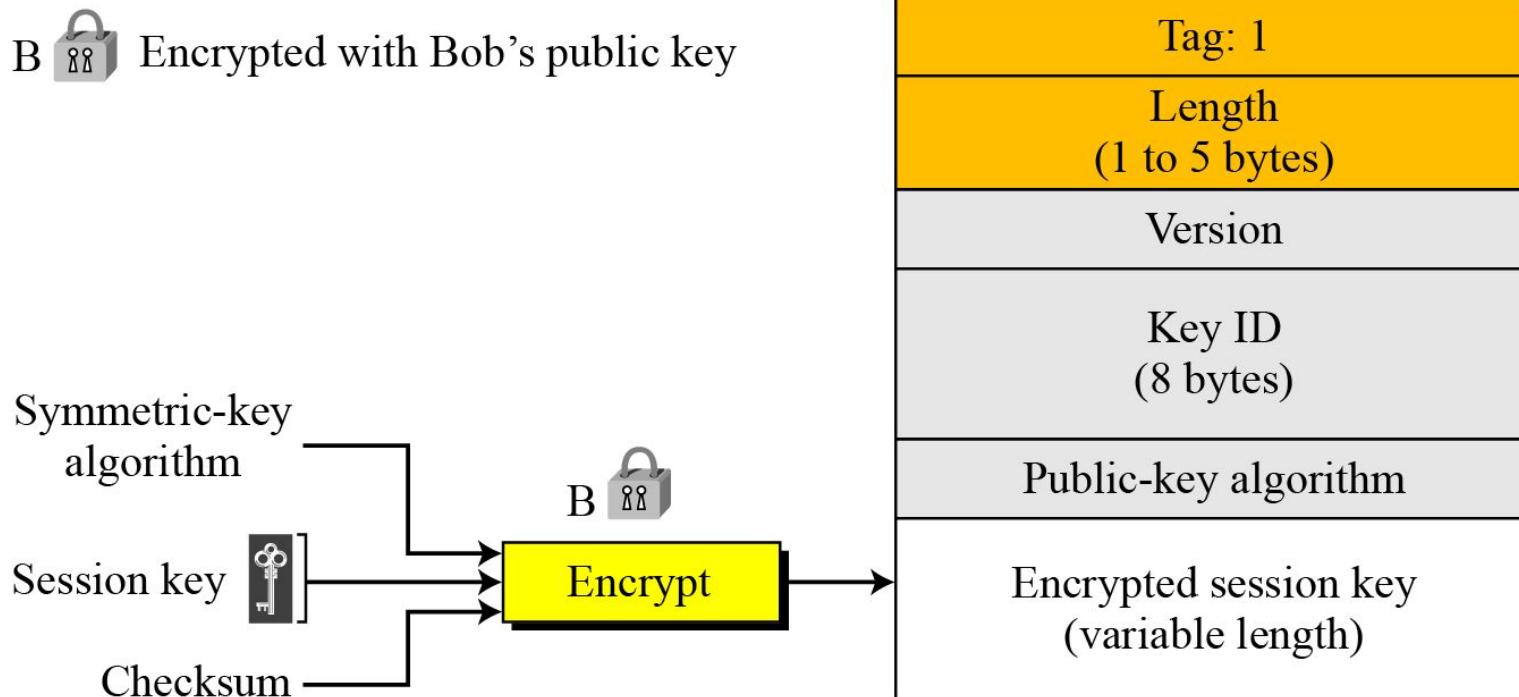
## 16.2.6 *Continued*

**Table 16.13** *Some signature values*

<i>Value</i>	<i>Signature</i>
0x00	Signature of a binary document (message or file).
0x01	Signature of a text document (message or file).
0x10	Generic certificate of a user ID and public-key packet. The signer does not make any particular assertion about the owner of the key.
0x11	Personal certificate of a user ID and public-key packet. No verification is done on the owner of the key.
0x12	Casual certificate of a User ID and public-key packet. Some casual verification done on the owner of the key.
0x13	Positive certificate of a user ID and public-key packet. Substantial verification done.
0x30	Certificate revocation signature. This removes an earlier certificate (0x10 through 0x13).

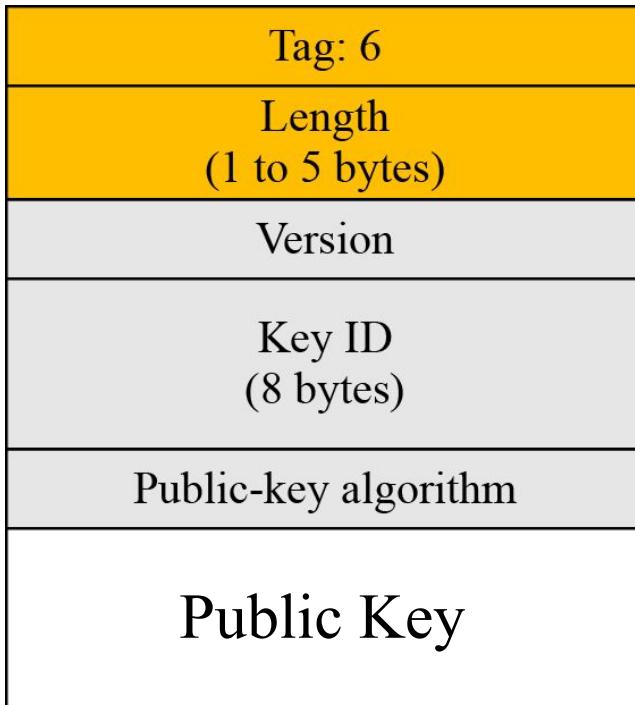
## 16.2.6 Continued

Figure 16.17 Session-key packet



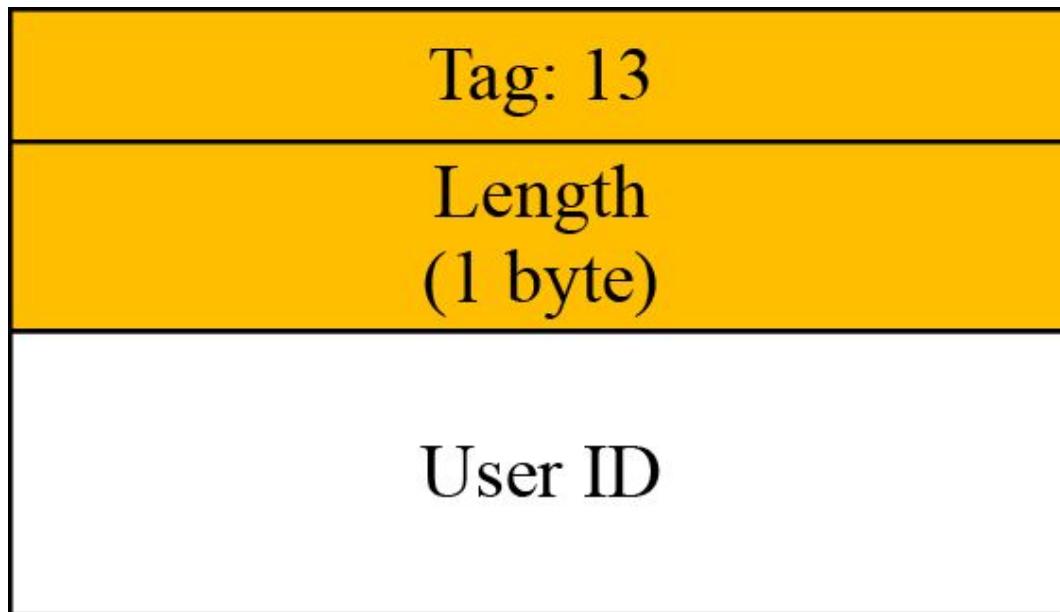
## 16.2.6 *Continued*

**Figure 16.18** *Public-key packet*



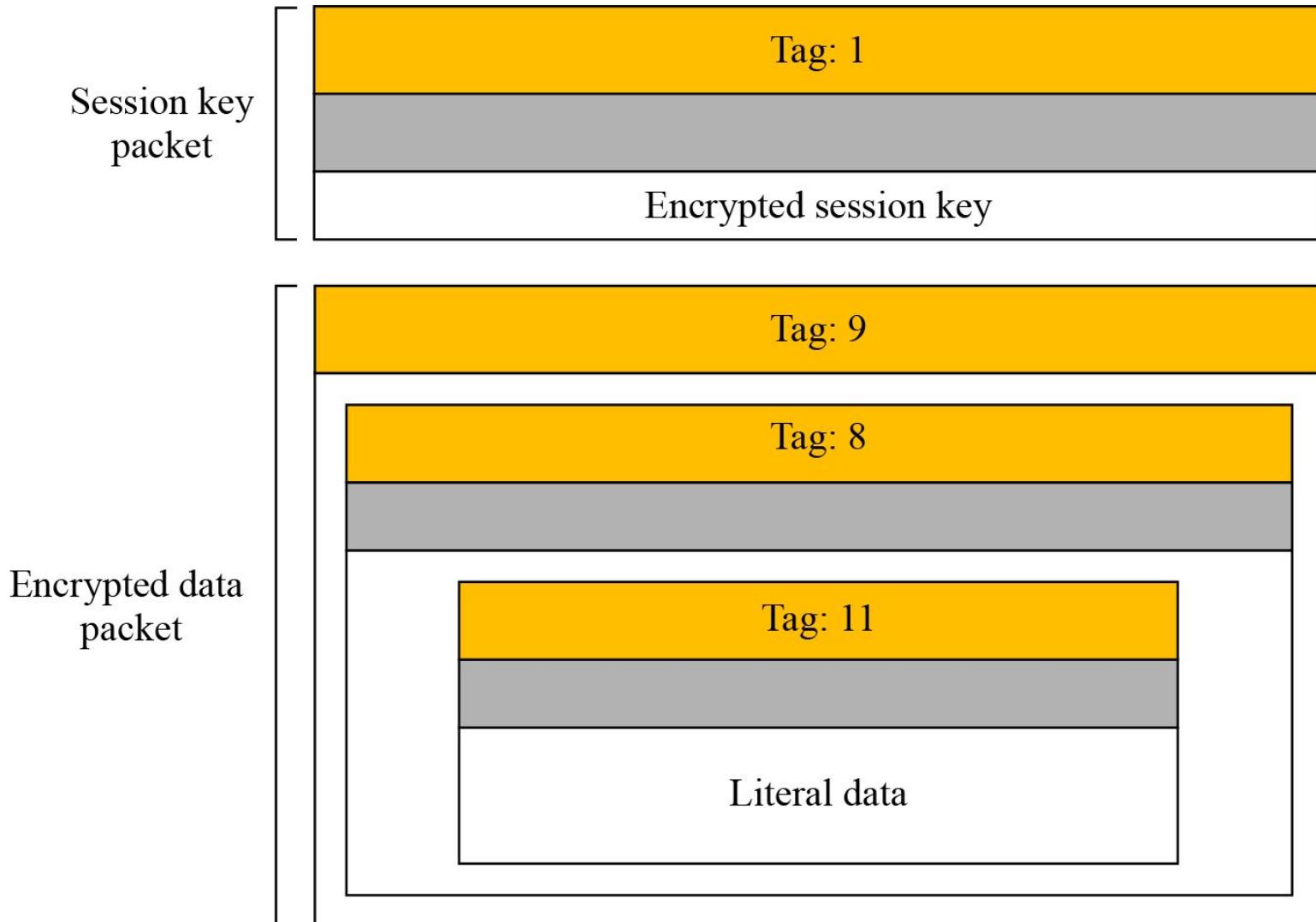
## 16.2.6 *Continued*

**Figure 16.19** *User ID packet*



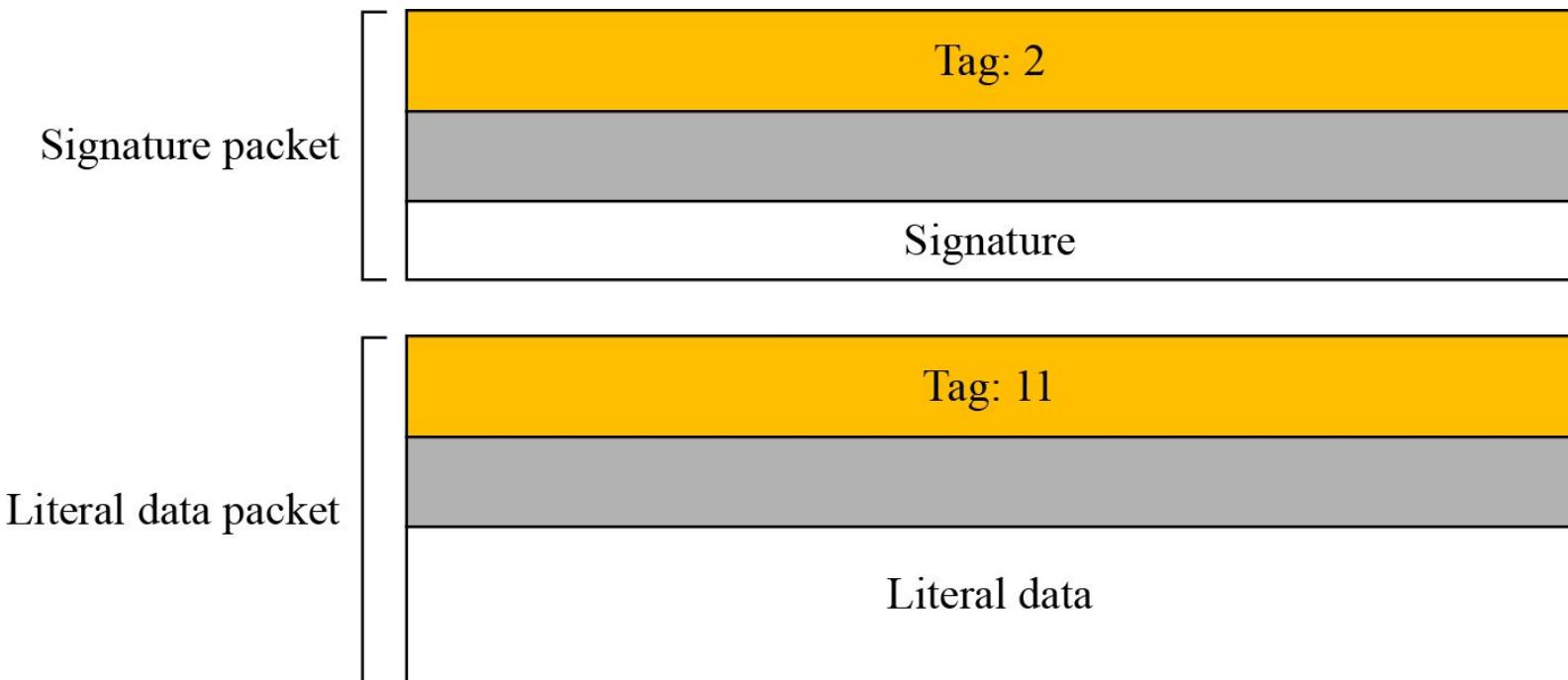
## 16.2.7 PGP Messages

Figure 16.20 *Encrypted message*



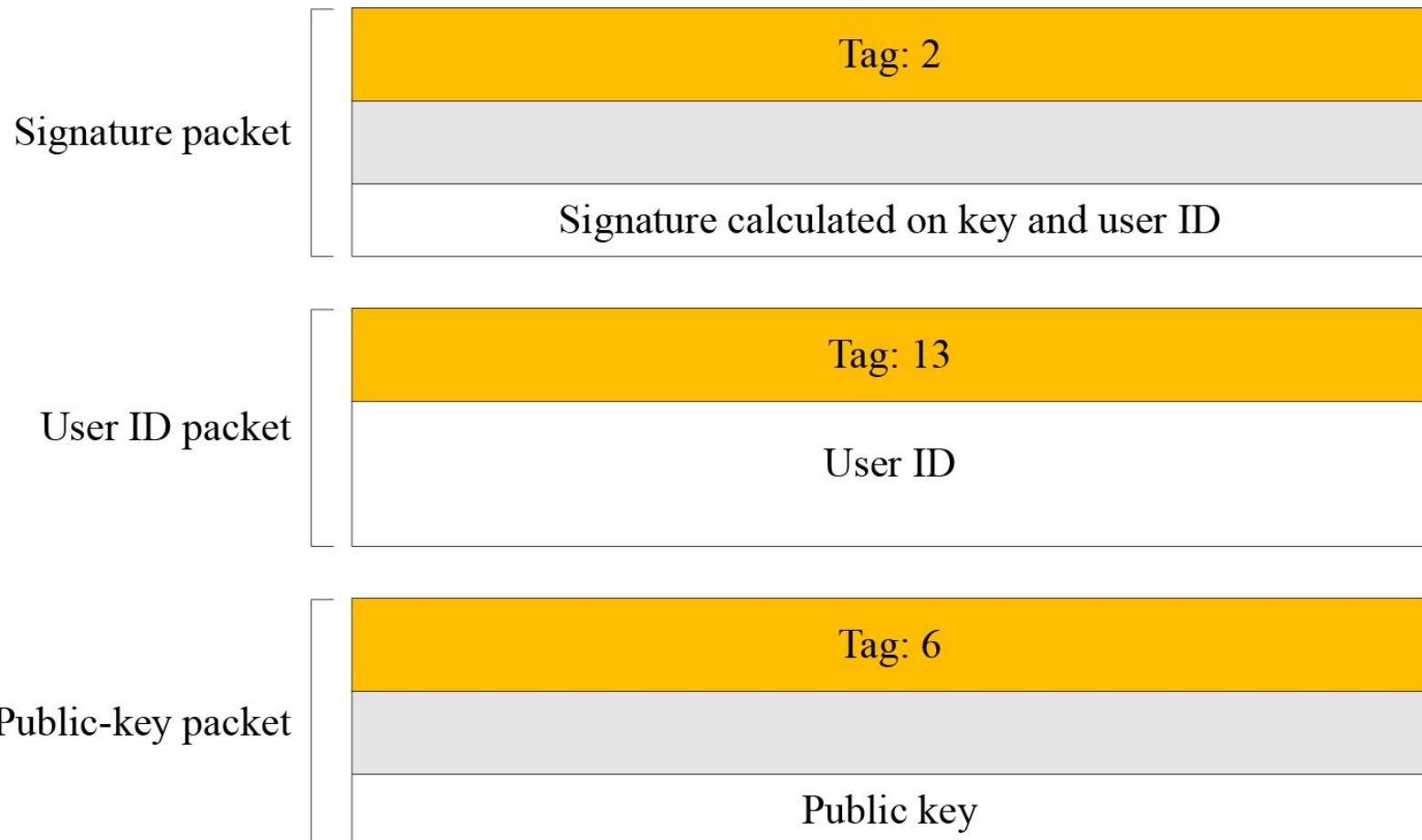
## 16.2.7 *Continued*

**Figure 16.21** *Signed message*



## 16.2.7 *Continued*

**Figure 16.22 Certificate message**



## 16-3 S/MIME

*Another security service designed for electronic mail is Secure/Multipurpose Internet Mail Extension (S/MIME). The protocol is an enhancement of the Multipurpose Internet Mail Extension (MIME) protocol.*

### *Topics discussed in this section:*

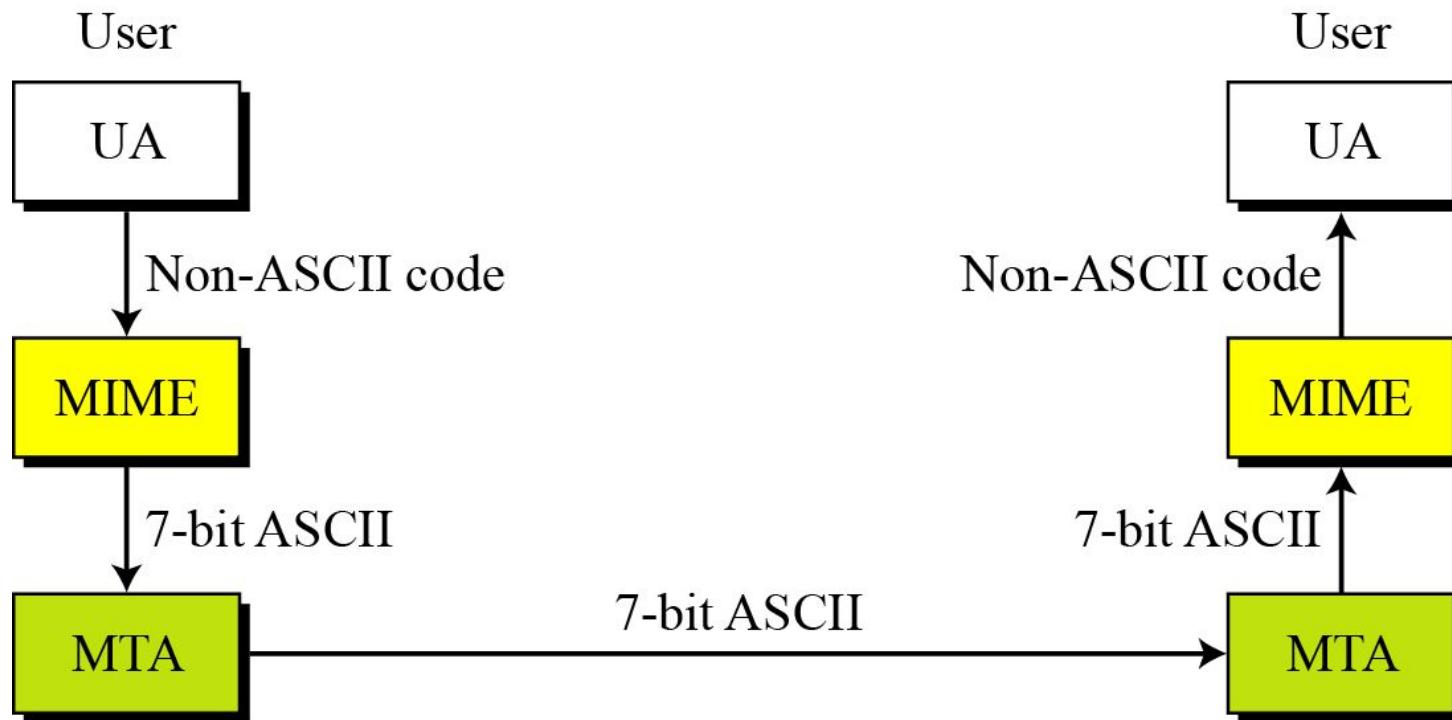
**16.3.1 MIME**

**16.3.2 S/MIME**

**16.3.3 Applications of S/MIME**

## 16.3.1 *Continued*

**Figure 16.23** *MIME*



## 16.3.1 *Continued*

**Figure 16.24 Teledesic**

E-mail header	MIME headers
MIME-Version: 1.1 Content-Type: type/subtype Content-Transfer-Encoding: encoding type Content-Id: message id Content-Description: textual explanation of nontextual contents	
E-mail body	

## 16.3.1 *Continued*

### *MIME-Version*

*This header defines the version of MIME used. The current version is 1.1.*

**MIME-Version: 1.1**

### *Content-Type*

*The content type and the content subtype are separated by a slash. Depending on the subtype, the header may contain other parameters.*

**Content-Type: <type / subtype; parameters>**

## 16.3.1 *Continued*

**Table 16.14** *Data types and subtypes in MIME*

Type	Subtype	Description
	Plain	Unformatted.
	HTML	HTML format.
Multipart	Mixed	Body contains ordered parts of different data types.
	Parallel	Same as above, but no order.
	Digest	Similar to Mixed, but the default is message/RFC822.
	Alternative	Parts are different versions of the same message.
Message	RFC822	Body is an encapsulated message.
	Partial	Body is a fragment of a bigger message.
	External-Body	Body is a reference to another message.
Image	JPEG	Image is in JPEG format.
	GIF	Image is in GIF format.
Video	MPEG	Video is in MPEG format.
Audio	Basic	Single channel encoding of voice at 8 KHz.
Application	PostScript	Adobe PostScript.
	Octet-stream	General binary data (eight-bit bytes).

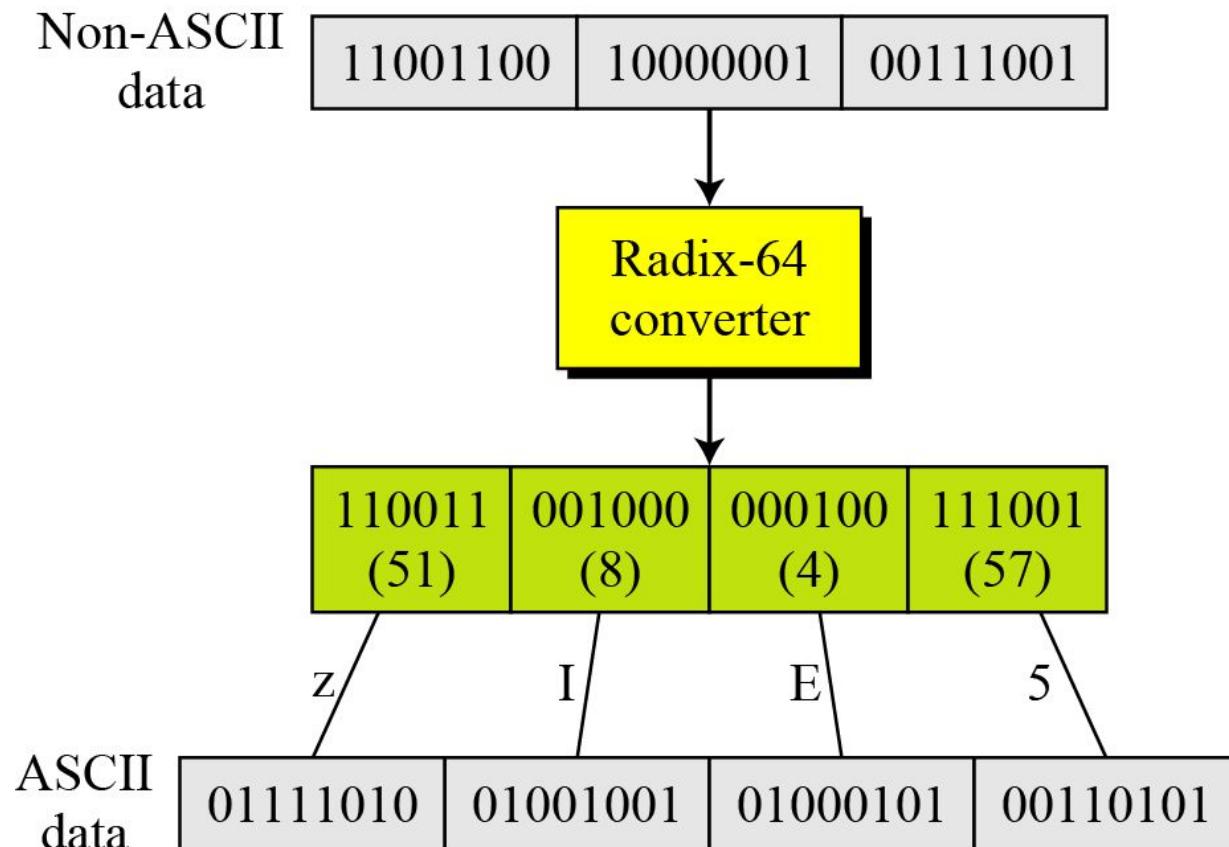
## 16.3.1 *Continued*

**Table 16.15** *Content-transfer-encoding*

Type	Description
7bit	NVT ASCII characters and short lines.
8bit	Non-ASCII characters and short lines.
Binary	Non-ASCII characters with unlimited-length lines.
Radix-64	6-bit blocks of data are encoded into 8-bit ASCII characters using Radix-64 conversion.
Quoted-printable	Non-ASCII characters are encoded as an equal sign followed by an ASCII code.

## 16.3.1 Continued

Figure 16.25 Radix-64 conversion



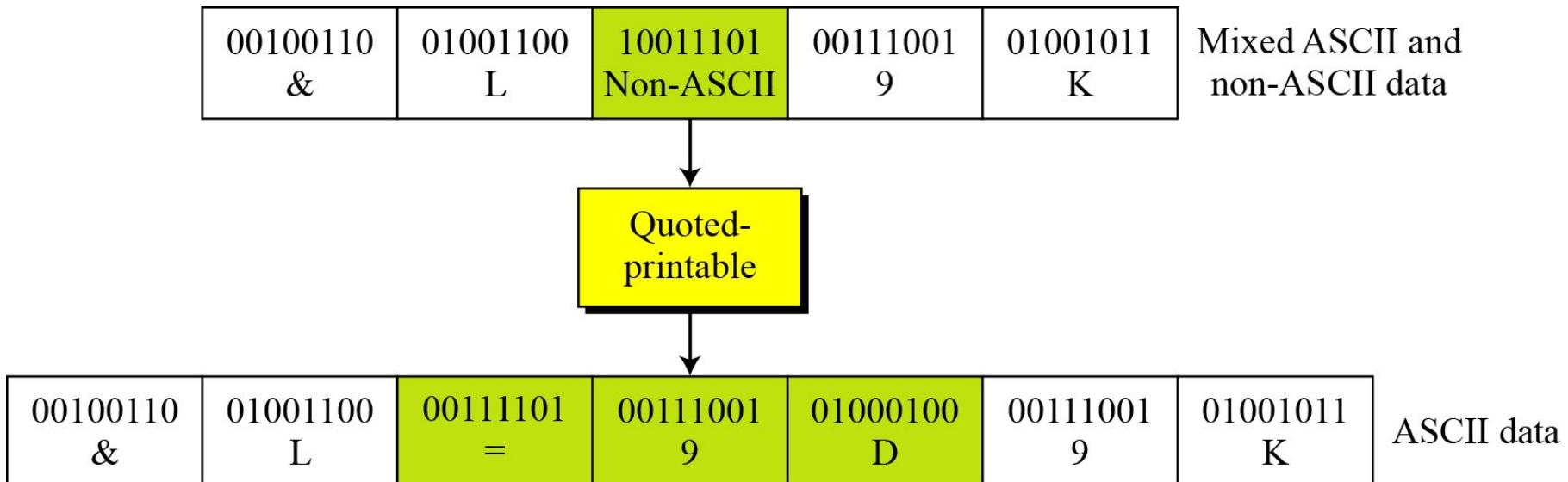
## 16.3.1 Continued

**Table 16.16** Radix-64 encoding table

Value	Code										
0	<b>A</b>	11	<b>L</b>	22	<b>W</b>	33	<b>h</b>	44	<b>s</b>	55	<b>3</b>
1	<b>B</b>	12	<b>M</b>	23	<b>X</b>	34	<b>i</b>	45	<b>t</b>	56	<b>4</b>
2	<b>C</b>	13	<b>N</b>	24	<b>Y</b>	35	<b>j</b>	46	<b>u</b>	57	<b>5</b>
3	<b>D</b>	14	<b>O</b>	25	<b>Z</b>	36	<b>k</b>	47	<b>v</b>	58	<b>6</b>
4	<b>E</b>	15	<b>P</b>	26	<b>a</b>	37	<b>l</b>	48	<b>w</b>	59	<b>7</b>
5	<b>F</b>	16	<b>Q</b>	27	<b>b</b>	38	<b>m</b>	49	<b>x</b>	60	<b>8</b>
6	<b>G</b>	17	<b>R</b>	28	<b>c</b>	39	<b>n</b>	50	<b>y</b>	61	<b>9</b>
7	<b>H</b>	18	<b>S</b>	29	<b>d</b>	40	<b>o</b>	51	<b>z</b>	62	<b>+</b>
8	<b>I</b>	19	<b>T</b>	30	<b>e</b>	41	<b>p</b>	52	<b>0</b>	63	<b>/</b>
9	<b>J</b>	20	<b>U</b>	31	<b>f</b>	42	<b>q</b>	53	<b>1</b>		
10	<b>K</b>	21	<b>V</b>	32	<b>g</b>	43	<b>r</b>	54	<b>2</b>		

## 16.3.1 *Continued*

**Figure 16.26 Quoted-printable**



## **16.3.2 S/MIME**

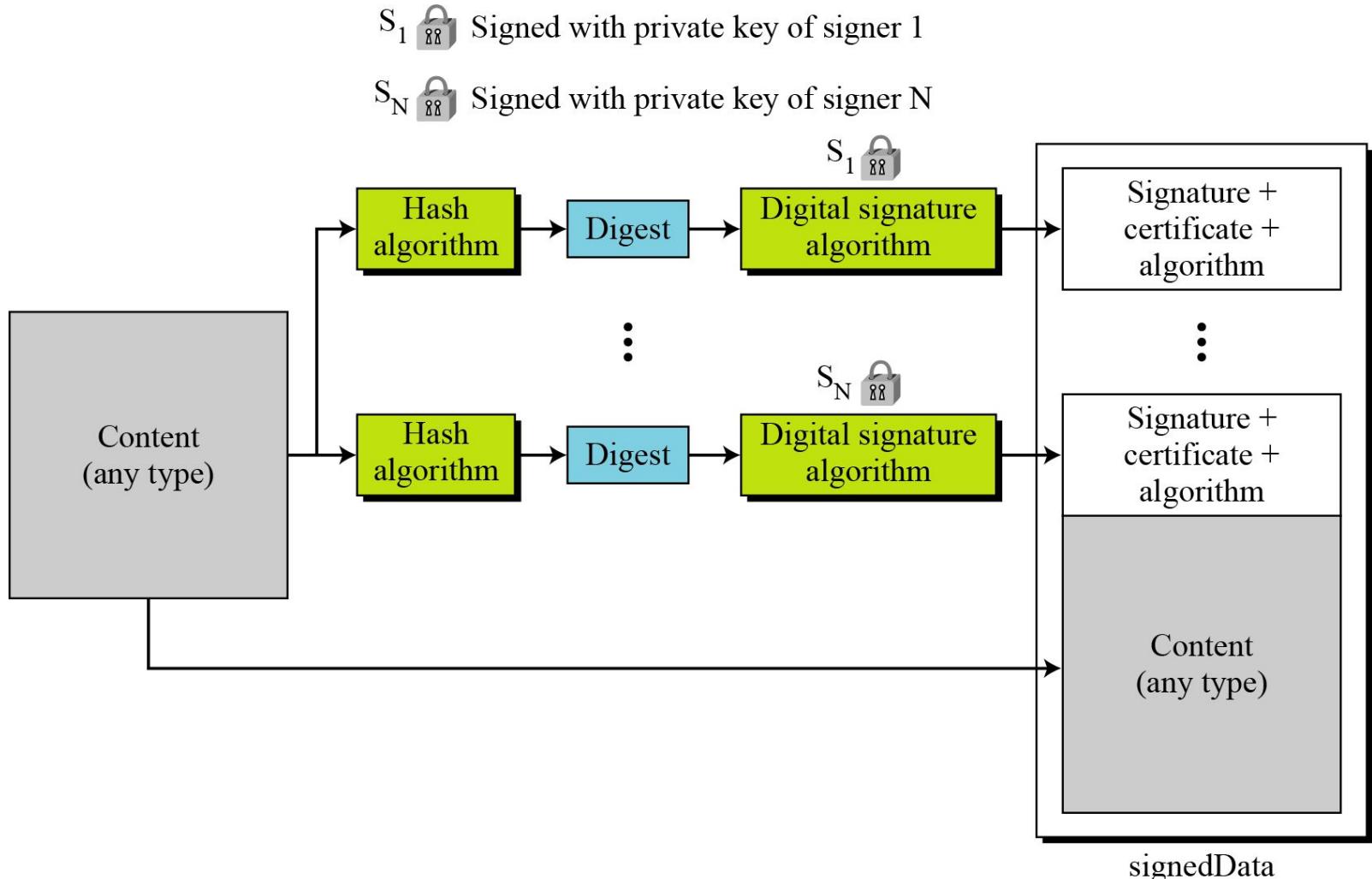
*S/MIME adds some new content types to include security services to the MIME. All of these new types include the parameter “application/pkcs7-mime,” in which “pkcs” defines “**Public Key Cryptography Specification.**”*

### **Cryptographic Message Syntax (CMS)**

*To define how security services, such as confidentiality or integrity, can be added to MIME content types, S/MIME has defined Cryptographic Message Syntax (CMS). The syntax in each case defines the exact encoding scheme for each content type. For details, the reader is referred to **RFC 3369** and **3370**.*

## 16.3.2 Continued

**Figure 16.27 Signed-data content type**



## 16.3.2 Continued

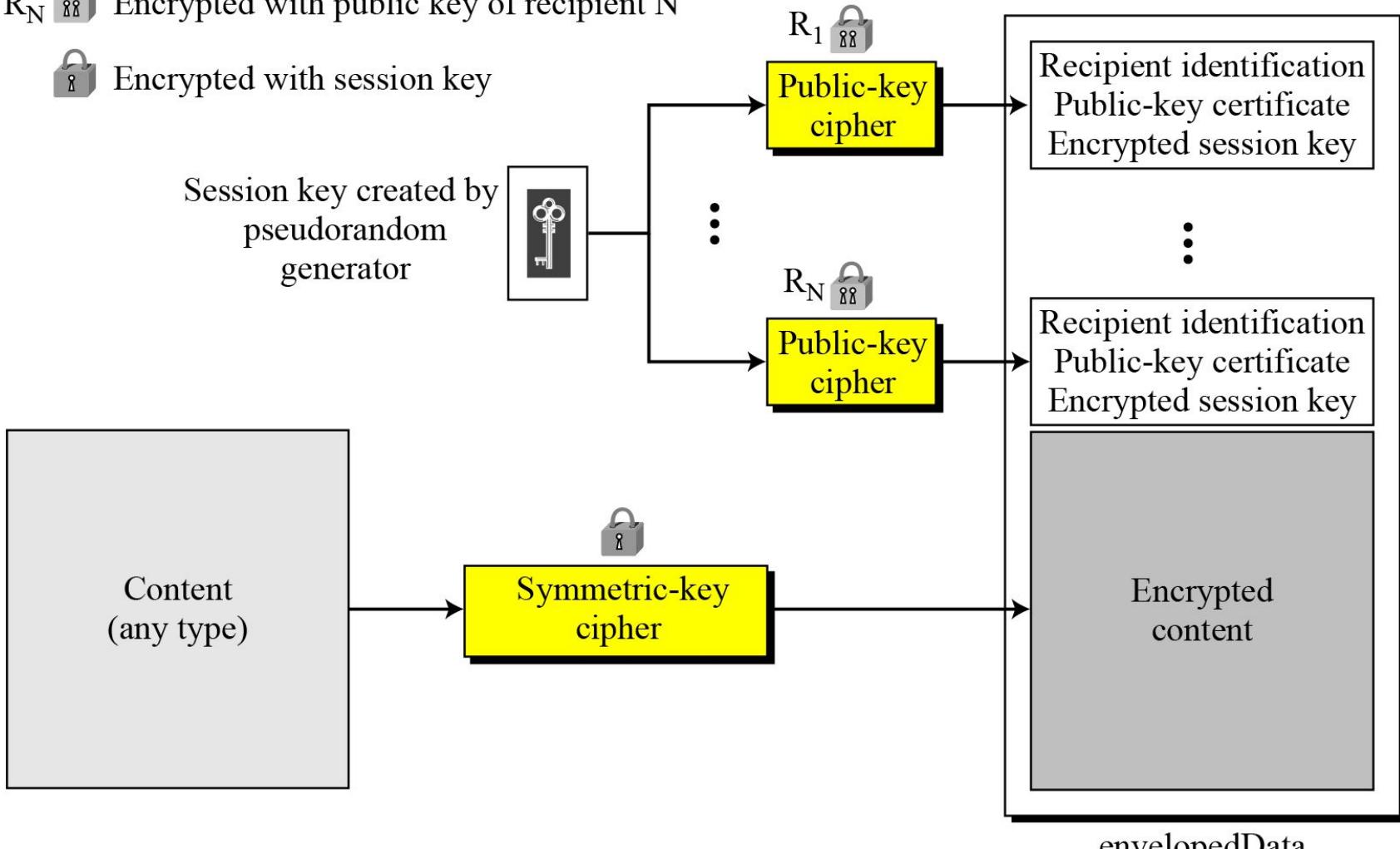
Figure 16.28 Enveloped-data content type

$R_1$  Encrypted with public key of recipient 1

$R_N$  Encrypted with public key of recipient N

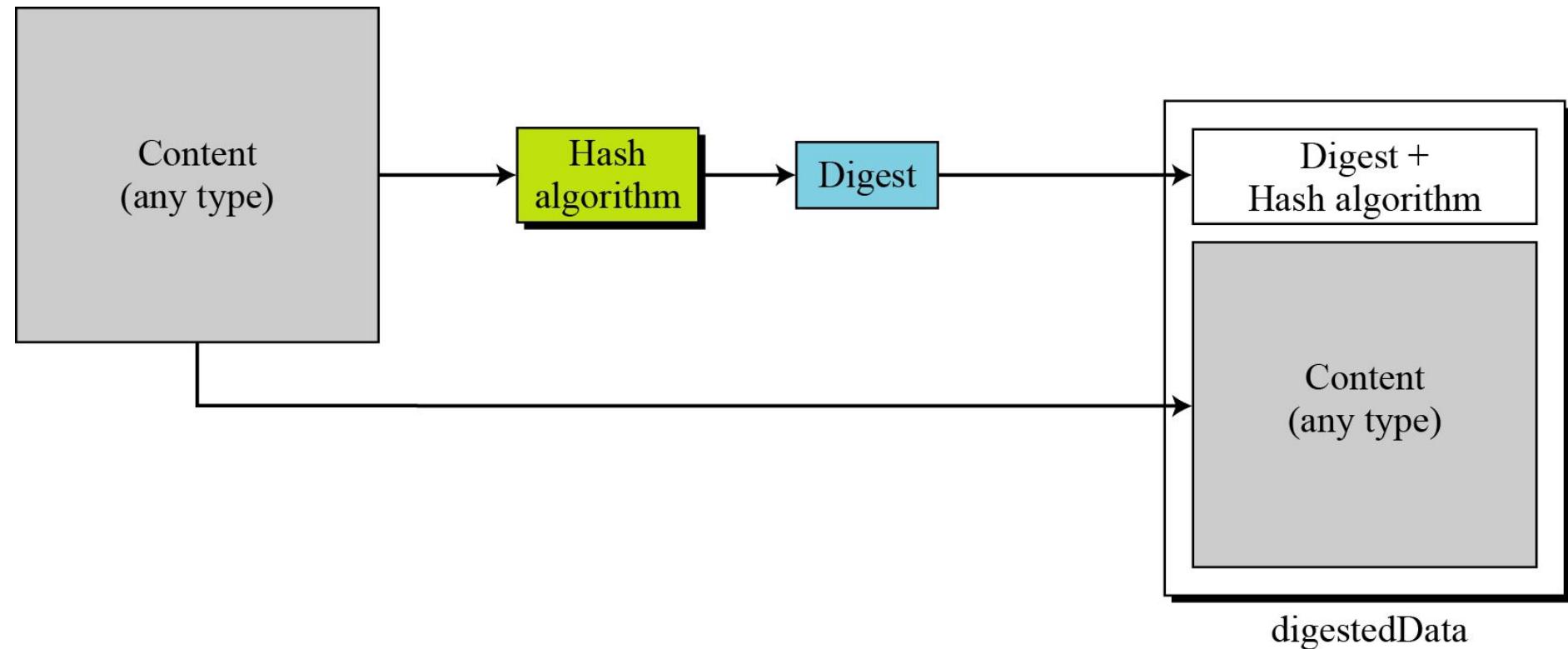
Encrypted with session key

Session key created by  
pseudorandom  
generator



## 16.3.2 *Continued*

Figure 16.29 *Digest-data content type*

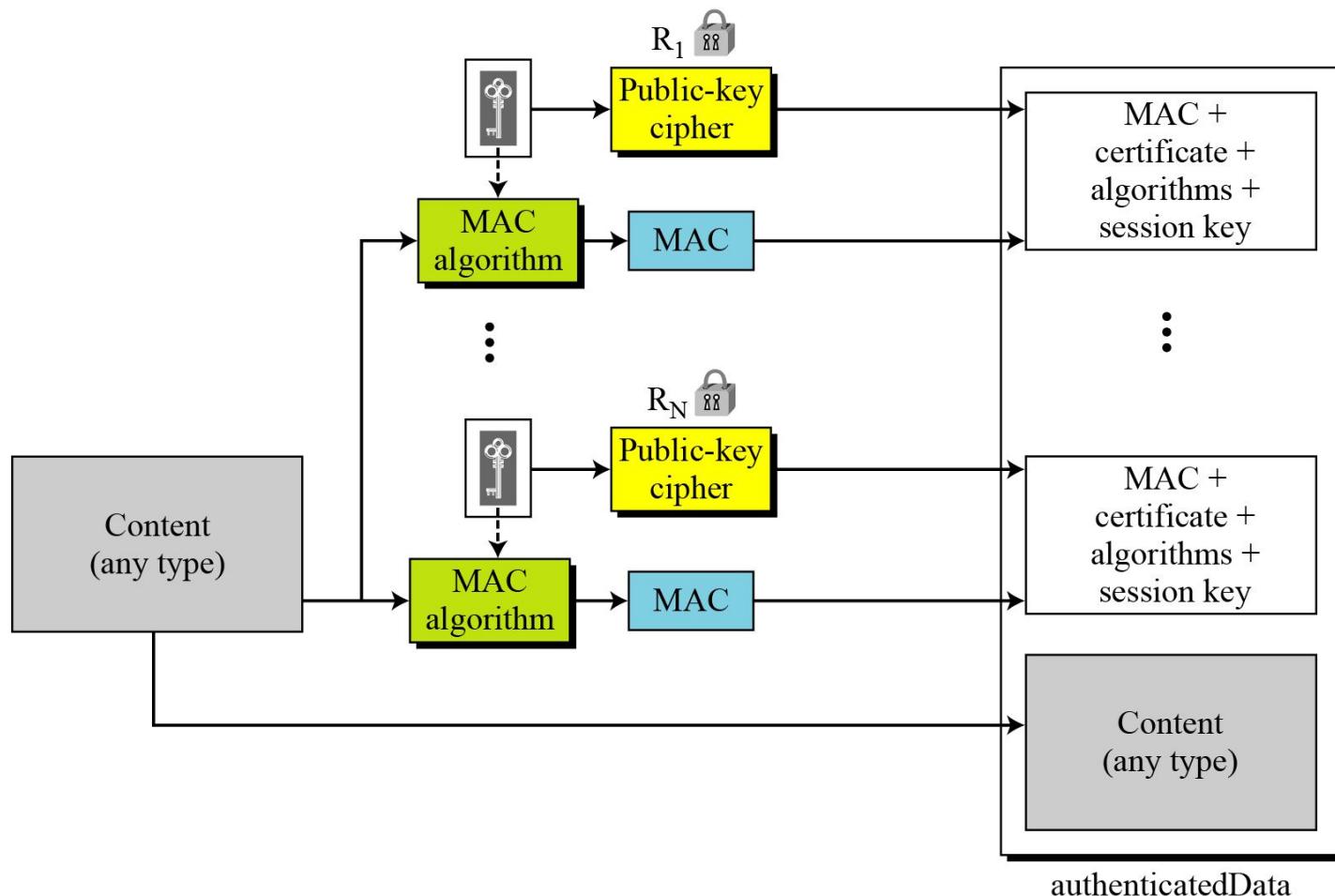


## 16.3.2 Continued

Figure 16.30 Authenticated-data content type

$R_1$  Encrypted with public key of recipient 1

$R_N$  Encrypted with public key of recipient N



## 16.3.2 Continued

### Cryptographic Algorithms

S/MIME defines several cryptographic algorithms. The term “**must**” means an absolute requirement; the term “**should**” means recommendation.

**Table 16.17** Cryptographic algorithm for S/MIME

Algorithm	Sender must support	Receiver must support	Sender should support	Receiver should support
Content-encryption algorithm	Triple DES	Triple DES		1. AES 2. RC2/40
Session-key encryption algorithm	RSA	RSA	Diffie-Hellman	Diffie-Hellman
Hash algorithm	SHA-1	SHA-1		MD5
Digest-encryption algorithm	DSS	DSS	RSA	RSA
Message-authentication algorithm		HMAC with SHA-1		

## 16.3.2 *Continued*

### Example 16.3

The following shows an example of an enveloped-data in which a small message is encrypted using triple DES.

**Content-Type: application/pkcs7-mime; mime-type=enveloped-data**

**Content-Transfer-Encoding: Radix-64**

**Content-Description: attachment**

**name="report.txt";**

cb32ut67f4bhijHU21oi87eryb0287hmnkls...  
ybmlkjhgfdyhGe23Kjk34XiuD678Es16se09jy76jHuytTMDcbnmlkjgfFdiuyu678543m0n3h  
G34un12P2454Hoi87e2ryb0H2MjN6KuyrlsgFD...  
DoY897fk923jljk1301XiuD6gh78EsUyT23y

# Firewalls

(Slides: From Internet Source)

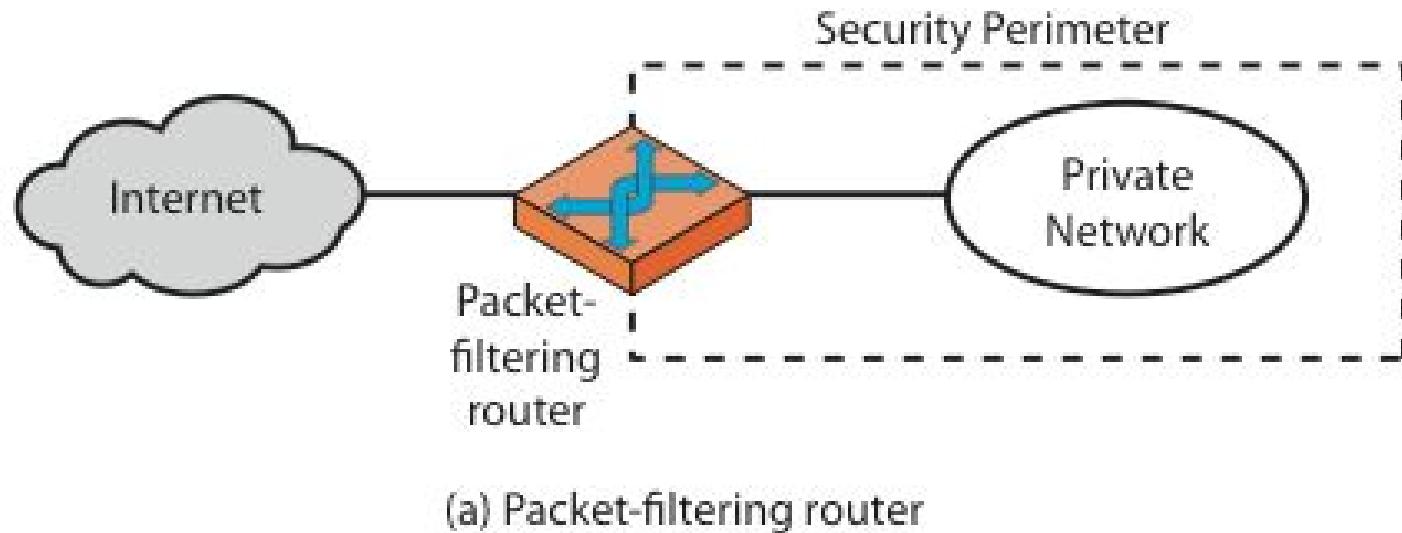
# Objectives and Deliverable

- Understand the concept of firewalls and the three major categories: packet filters (stateless vs. stateful) and application level proxy
- Can apply them to detect different attacks
- Be able to write the stateless packet filter rules

# What is a Firewall?

- A **Cyber Security Tool** to filter traffic.
- Interconnects networks with differing trust
- Goal of a firewall is to block malicious traffic requests and data packets while allowing legitimate traffic through.
- Imposes restrictions on network services
  - only authorized traffic is allowed
- Auditing and controlling access
  - can implement alarms for abnormal behavior
- Itself immune to penetration

# Firewalls – Packet Filters



# Classification of Firewall

Characterized by protocol level it controls in

- Packet filtering
- Circuit gateways
- Application gateways

# Firewalls – Packet Filters

- Simplest of components
- Uses transport-layer information only
  - IP Source Address, Destination Address
  - Protocol/Next Header (TCP, UDP, ICMP, etc)
  - TCP or UDP source & destination ports
  - TCP Flags (SYN, ACK, FIN, etc)
  - ICMP message type
- Examples
  - DNS uses port 53
    - No incoming port 53 packets except known trusted servers

# Usage of Packet Filters

- Filtering with incoming or outgoing interfaces
  - E.g., Egress filtering of spoofed IP addresses
  - Ingress filtering
- Permits or denies certain services
  - Requires intimate knowledge of TCP and UDP port utilization on a number of operating systems

# How to Configure a Packet Filter

- Start with a security policy
- Specify allowable packets in terms of logical expressions on packet fields
- Rewrite expressions in syntax supported by your vendor
- General rules - least privilege
  - All that is not expressly permitted is prohibited
  - If you do not need it, eliminate it

Every ruleset is followed by an implicit rule  
reading like this.

action	src	port	dest	port	comment
block	*	*	*	*	<i>default</i>

## Example 1:

Suppose we want to allow inbound mail (SMTP, port 25) but only to our gateway machine. Also suppose that traffic from some particular site SPIGOT is to be blocked.

action	src	port	dest	port	comment
block	SPIGOT	*	*	*	<i>we don't trust these people</i>
allow	*	*	OUR-GW	25	<i>connection to our SMTP port</i>

## Example 2:

Now suppose that we want to implement the policy “any inside host can send mail to the outside”.

action	ourhost	port	theirhost	port	comment
allow	*	*	*	25	<i>connection to their SMTP port</i>

# Security & Performance of Packet Filters

- Degradation depends on number of rules applied at any point
- Order rules so that most common traffic is dealt with first
- Correctness is more important than speed

Rule	Source Address	Destination Address	Action	Comments	
R1	111.11/16	222.22.22/24	permit	Let datagrams from Bob's university network into a restricted subnet.	
R2	111.11.11/24	222.22/16	deny	Don't let traffic from Trudy's subnet into anywhere within Alice's network.	
R3	0.0.0.0/0	0.0.0.0/0	deny	Don't let traffic into Alice's network.	
Datagram Number	Source IP Address	Destination IP Address	Desired Action	Action Under R2, R1, R3	Action Under R1, R2, R3
P1	111.11.11.1 (hacker subnet)	222.22.6.6 (corp.net)			
P2	111.11.11.1 (hacker subnet)	222.22.22.2 (special subnet)			
P3	111.11.6.6 (univ. net, not the hacker subnet)	222.22.22.2 (special subnet)			
P4	111.11.6.6 (univ. net, not the hacker subnet)	222.22.6.6 (corp. net)			

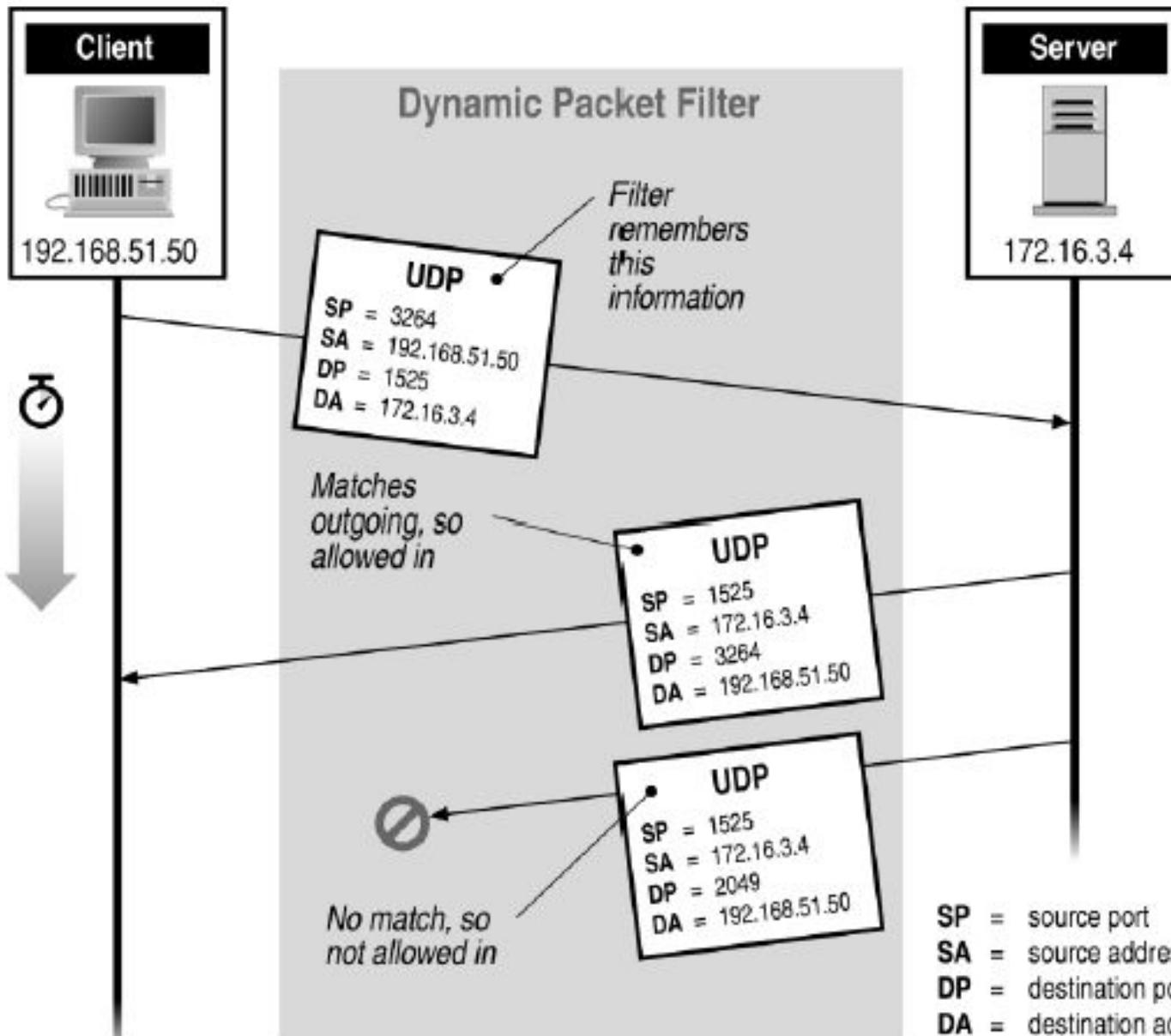
# Port Numbering

- TCP connection
  - Server port is number less than 1024
  - Client port is number between 1024 and 16383
- Permanent assignment
  - Ports <1024 assigned permanently
    - 20,21 for FTP
    - 23 for Telnet
    - 25 for server SMTP
    - 80 for HTTP
- Variable use
  - Ports >1024 must be available for client to make any connection
  - This presents a limitation for stateless packet filtering
    - If client wants to use port 2048, firewall must allow *incoming* traffic on this port
  - Better: stateful filtering knows outgoing requests

# Firewalls – Stateful Packet Filters

- Traditional packet filters do not examine transport layer context
  - ie matching return packets with outgoing flow
- Stateful packet filters address this need
- They examine each IP packet in context
  - Keep track of client-server sessions
  - Check each packet validly belongs to one
- Hence are better able to detect bogus packets out of context

# Stateful Filtering



## Example 2(Revisited):

Now suppose that we want to implement the policy “any inside host can send mail to the outside”.

action	ourhost	port	theirhost	port	comment
allow	*	*	*	25	<i>connection to their SMTP port</i>

This solution allows calls to come from any port on an inside machine, and will direct them to port 25 on the outside. Simple enough...

So why is it wrong?

- Our defined restriction is based solely on the outside host's port number, which we have no way of controlling.
- Now an enemy can access any internal machines and port by originating his call from port 25 on the outside machine.

What can be a better solution ?

action	src	port	dest	port	flags	comment
allow	{our hosts}	*	*	25		<i>our packets to their SMTP port</i>
allow	*	25	*	*	ACK	<i>their replies</i>

- The ACK signifies that the packet is part of an ongoing conversation
- Packets without the ACK are connection establishment messages, which we are only permitting from internal hosts

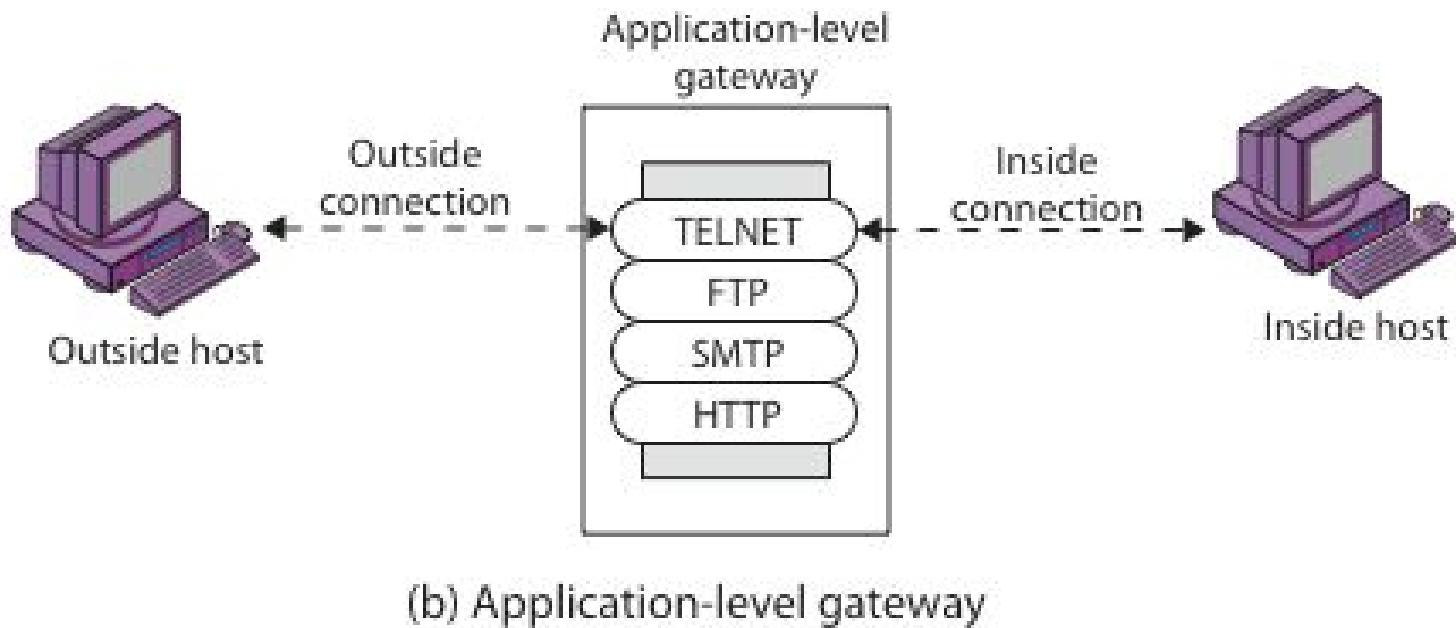
# Firewall Outlines

- Packet filtering
- Application gateways
- Circuit gateways

# Firewall Gateways

- Firewall runs set of proxy programs
  - Proxies filter incoming, outgoing packets
  - All incoming traffic directed to firewall
  - All outgoing traffic appears to come from firewall
- Policy embedded in proxy programs
- Two kinds of proxies
  - Application-level gateways/proxies
    - Tailored to http, ftp, smtp, etc.
  - Circuit-level gateways/proxies
    - Working on TCP level

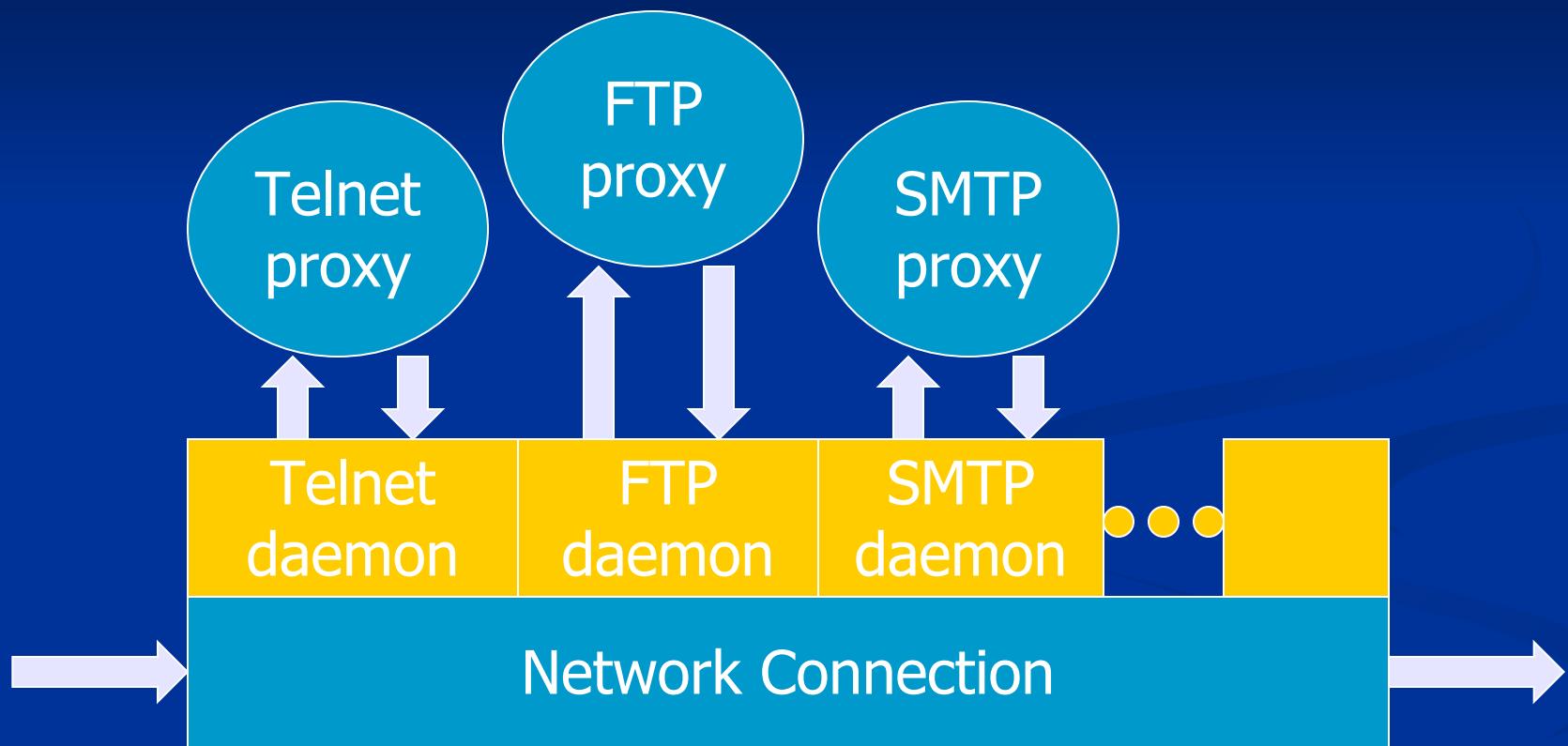
# Firewalls - Application Level Gateway (or Proxy)



# Application-Level Filtering

- Has full access to protocol
  - user requests service from proxy
  - proxy validates request as legal
  - then actions request and returns result to user
- Need separate proxies for each service
  - E.g., SMTP (E-Mail)
  - TELNET (Remote Login)
  - DNS (Domain Name System)
  - FTP (File Transfer)

# App-level Firewall Architecture



Daemon spawns proxy when communication detected

# Quiz

- In this question, we explore some applications and limitations of a packet filtering firewall. For each of the question, briefly explain 1) can stateless firewall be configured to defend against the attack and how?  
2) if not, what about stateful firewall ?  
3) if neither can, what about application-level proxy?
  - Can the firewall prevent a SYN flood attack from the external network?
  - Can the firewall prevent a Smurf attack from the external network? Basically, the Smurf attack uses the broadcast IP address of the subnet.

GLOBAL  
EDITION

# Cryptography and Network Security

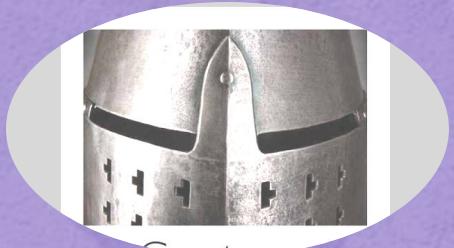
*Principles and Practice*

SEVENTH EDITION

William Stallings



Pearson



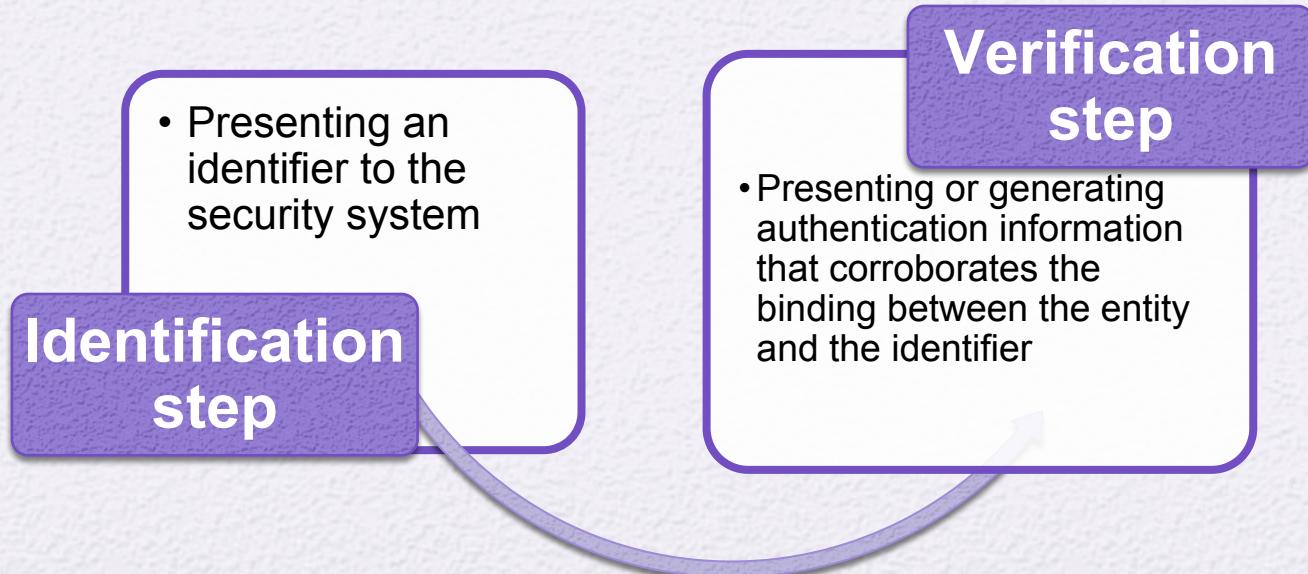
Chapter 15

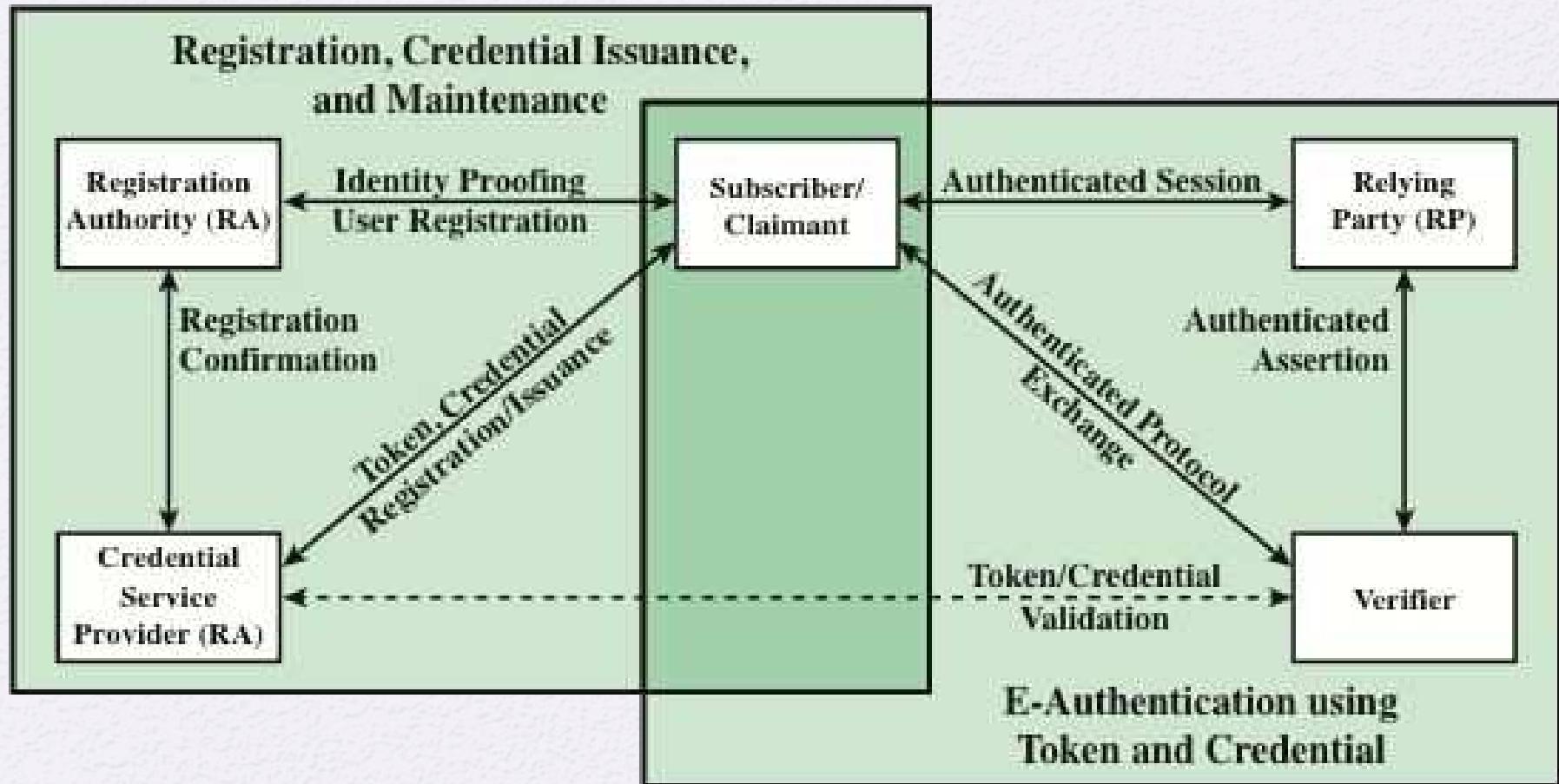
# Chapter 15

## User Authentication

# Remote User-Authentication Principles

- The process of verifying an identity claimed by or for a system entity
- An authentication process consists of two steps:





**Figure 15.1 The NIST SP 800-63-2 E-Authentication Architectural Model**

# Means of User Authentication

- Examples include recognition by fingerprint, retina, and face
  - Examples include a personal password, a PIN, or a set of answers to a rearranged question.
  - Examples include a password, a personal identifier, a number on a card, or a PIN, or a set of answers to a rearranged question.
  - Examples include cryptographic keys, electronic keyboards, smart cards, and physical keys referred to as a token.
- Something the individual knows**
- Something the individual is (static biometrics)**
- Something the individual does (dynamic biometrics)**
- Something the individual has**
- Something the individual possesses**
- There are four general means of authenticating a user's identity, which can be used alone or in combination**
- For network-based user authentication, the most important methods involve cryptographic keys and something the individual knows, such as a password

# Mutual Authentication

- Protocols which enable communicating parties to satisfy themselves mutually about each other's identity and to exchange session keys

Central to the problem of authenticated key exchange are two issues:

## Timeliness

- Important because of the threat of message replays
- Such replays could allow an opponent to:
  - compromise a session key
  - successfully impersonate another party
  - disrupt operations by presenting parties with messages that appear genuine but are not

## Confidentiality

- Essential identification and session-key information must be communicated in encrypted form
- This requires the prior existence of secret or public keys that can be used for this purpose

# Replay Attacks

1. The simplest replay attack is one in which the opponent simply copies a message and replays it later
2. An opponent can replay a timestamped message within the valid time window
3. An opponent can replay a timestamped message within the valid time window, but in addition, the opponent suppresses the original message; thus, the repetition cannot be detected
4. Another attack involves a backward replay without modification and is possible if symmetric encryption is used and the sender cannot easily recognize the difference between messages sent and messages received on the basis of content

# Approaches to Coping With Replay Attacks

- Attach a sequence number to each message used in an authentication exchange
  - A new message is accepted only if its sequence number is in the proper order
  - Difficulty with this approach is that it requires each party to keep track of the last sequence number for each claimant it has dealt with
  - Generally not used for authentication and key exchange because of overhead
- Timestamps
  - Requires that clocks among the various participants be synchronized
  - Party A accepts a message as fresh only if the message contains a timestamp that, in A's judgment, is close enough to A's knowledge of current time
- Challenge/response
  - Party A, expecting a fresh message from B, first sends B a nonce (challenge) and requires that the subsequent message (response) received from B contain the correct nonce value

# One-Way Authentication

**One application for which encryption is growing in popularity is electronic mail (e-mail)**

- Header of the e-mail message must be in the clear so that the message can be handled by the store-and-forward e-mail protocol, such as SMTP or X.400
- The e-mail message should be encrypted such that the mail-handling system is not in possession of the decryption key

**A second requirement is that of authentication**

- The recipient wants some assurance that the message is from the alleged sender

# Remote User-Authentication Using Symmetric Encryption

**A two-level hierarchy of symmetric keys can be used to provide confidentiality for communication in a distributed environment**

- Strategy involves the use of a trusted key distribution center (KDC)
- Each party shares a secret key, known as a master key, with the KDC
- KDC is responsible for generating keys to be used for a short time over a connection between two parties and for distributing those keys using the master keys to protect the distribution

# Suppress-Replay Attacks

- The Denning protocol requires reliance on clocks that are synchronized throughout the network
- A risk involved is based on the fact that the distributed clocks can become unsynchronized as a result of sabotage or faults in the clocks or the synchronization mechanism
- The problem occurs when a sender's clock is ahead of the intended recipient's clock
  - An opponent can intercept a message from the sender and replay it later when the timestamp in the message becomes current at the recipient's site
  - Such attacks are referred to as *suppress-replay attacks*

# Kerberos

- Authentication service developed as part of Project Athena at MIT
- A workstation cannot be trusted to identify its users correctly to network services
  - A user may gain access to a particular workstation and pretend to be another user operating from that workstation
  - A user may alter the network address of a workstation so that the requests sent from the altered workstation appear to come from the impersonated workstation
  - A user may eavesdrop on exchanges and use a replay attack to gain entrance to a server or to disrupt operations
- Kerberos provides a centralized authentication server whose function is to authenticate users to servers and servers to users
  - Relies exclusively on symmetric encryption, making no use of public-key encryption

# Kerberos Requirements

- The first published report on Kerberos listed the following requirements:
  - **Secure**
    - A network eavesdropper should not be able to obtain the necessary information to impersonate a user
  - **Reliable**
    - Should be highly reliable and should employ a distributed server architecture with one system able to back up another
  - **Transparent**
    - Ideally, the user should not be aware that authentication is taking place beyond the requirement to enter a password
  - **Scalable**
    - The system should be capable of supporting large numbers of clients and servers

# Kerberos Version 4

- Makes use of DES to provide the authentication service
- Authentication server (AS)
  - Knows the passwords of all users and stores these in a centralized database
  - Shares a unique secret key with each server
- Ticket
  - Created once the AS accepts the user as authentic; contains the user's ID and network address and the server's ID
  - Encrypted using the secret key shared by the AS and the server
- Ticket-granting server (TGS)
  - Issues tickets to users who have been authenticated to AS
  - Each time the user requires access to a new service the client applies to the TGS using the ticket to authenticate itself
  - The TGS then grants a ticket for the particular service
  - The client saves each service-granting ticket and uses it to authenticate its user to a server each time a particular service is requested

# The Version 4 Authentication Dialogue

The lifetime associated with the ticket-granting ticket creates a problem:

- If the lifetime is very short (e.g., minutes), the user will be repeatedly asked for a password
- If the lifetime is long (e.g., hours), then an opponent has a greater opportunity for replay

A network service (the TGS or an application service) must be able to prove that the person using a ticket is the same person to whom that ticket was issued

Servers need to authenticate themselves to users

Table 15.1 (page 470 in textbook)

# Summary of Kerberos Version 4 Message Exchanges

(1)  $C \rightarrow AS \quad ID_c \parallel ID_{tgt} \parallel TS_1$

(2)  $AS \rightarrow C \quad E(K_c, [K_{c,tgt} \parallel ID_{tgt} \parallel TS_2 \parallel Lifetime_2 \parallel Ticket_{tgt}])$

$$Ticket_{tgt} = E(K_{tgt}, [K_{c,tgt} \parallel ID_C \parallel AD_C \parallel ID_{tgt} \parallel TS_2 \parallel Lifetime_2])$$

(a) Authentication Service Exchange to obtain ticket-granting ticket

(3)  $C \rightarrow TGS \quad ID_v \parallel Ticket_{tgt} \parallel Authenticator_c$

(4)  $TGS \rightarrow C \quad E(K_{tgt}, [K_{c,v} \parallel ID_v \parallel TS_4 \parallel Ticket_v])$

$$Ticket_v = E(K_{tgt}, [K_{c,v} \parallel ID_C \parallel AD_C \parallel ID_{tgt} \parallel TS_2 \parallel Lifetime_2])$$

$$Ticket_v = E(K_v, [K_{c,v} \parallel ID_C \parallel AD_C \parallel ID_v \parallel TS_4 \parallel Lifetime_4])$$

$$Authenticator_c = E(K_{tgt}, [ID_C \parallel AD_C \parallel TS_3])$$

(b) Ticket-Granting Service Exchange to obtain service-granting ticket

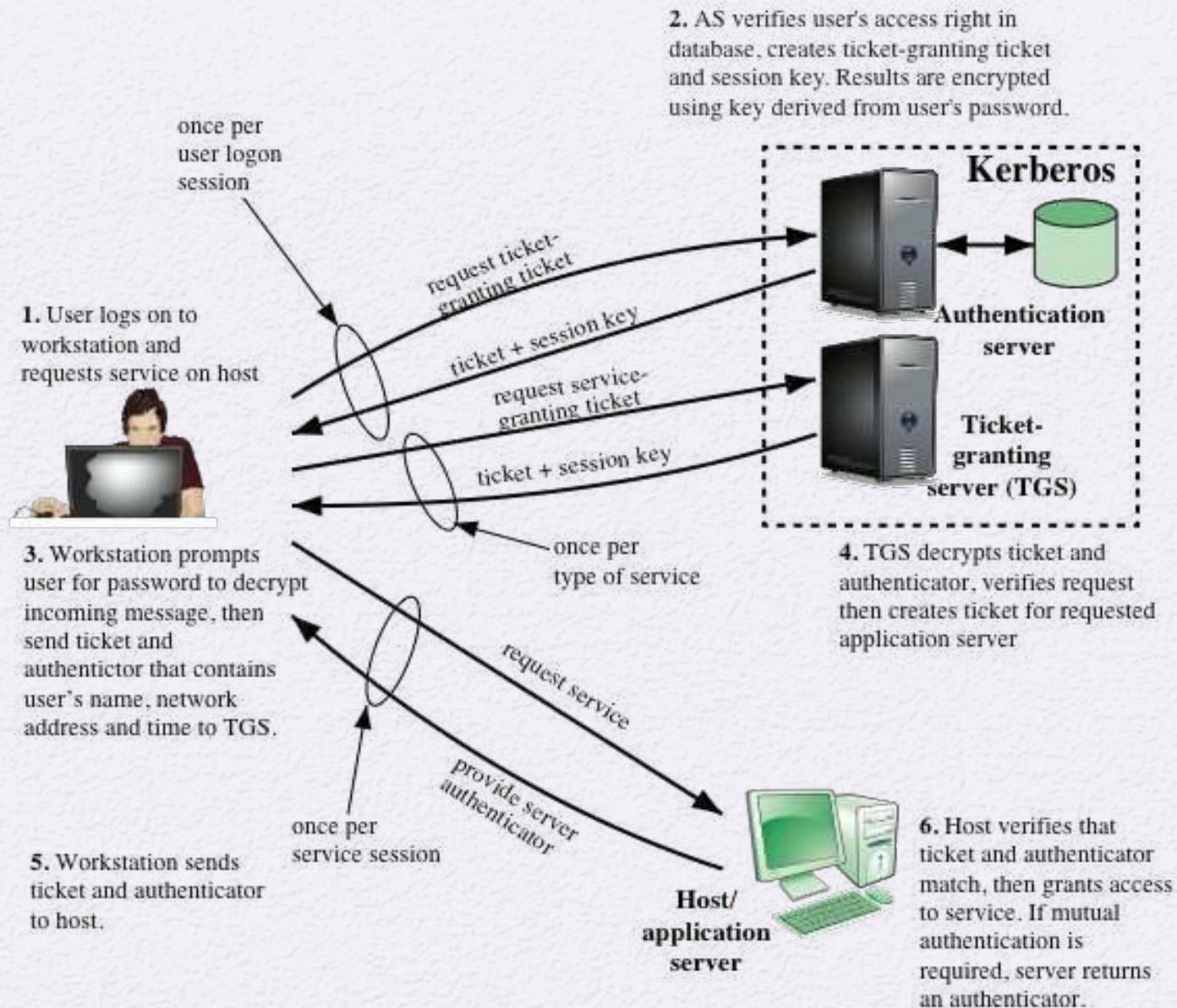
(5)  $C \rightarrow V \quad Ticket_v \parallel Authenticator_c$

(6)  $V \rightarrow C \quad E(K_{cv}, [TS_5 + 1])$  (for mutual authentication)

$$Ticket_v = E(K_v, [K_{cv} \parallel ID_C \parallel AD_C \parallel ID_v \parallel TS_4 \parallel Lifetime_4])$$

$$Authenticator_c = E(K_{cv}, [ID_C \parallel AD_C \parallel TS_5])$$

(c) Client/Server Authentication Exchange to obtain service



**Figure 15.2 Overview of Kerberos**

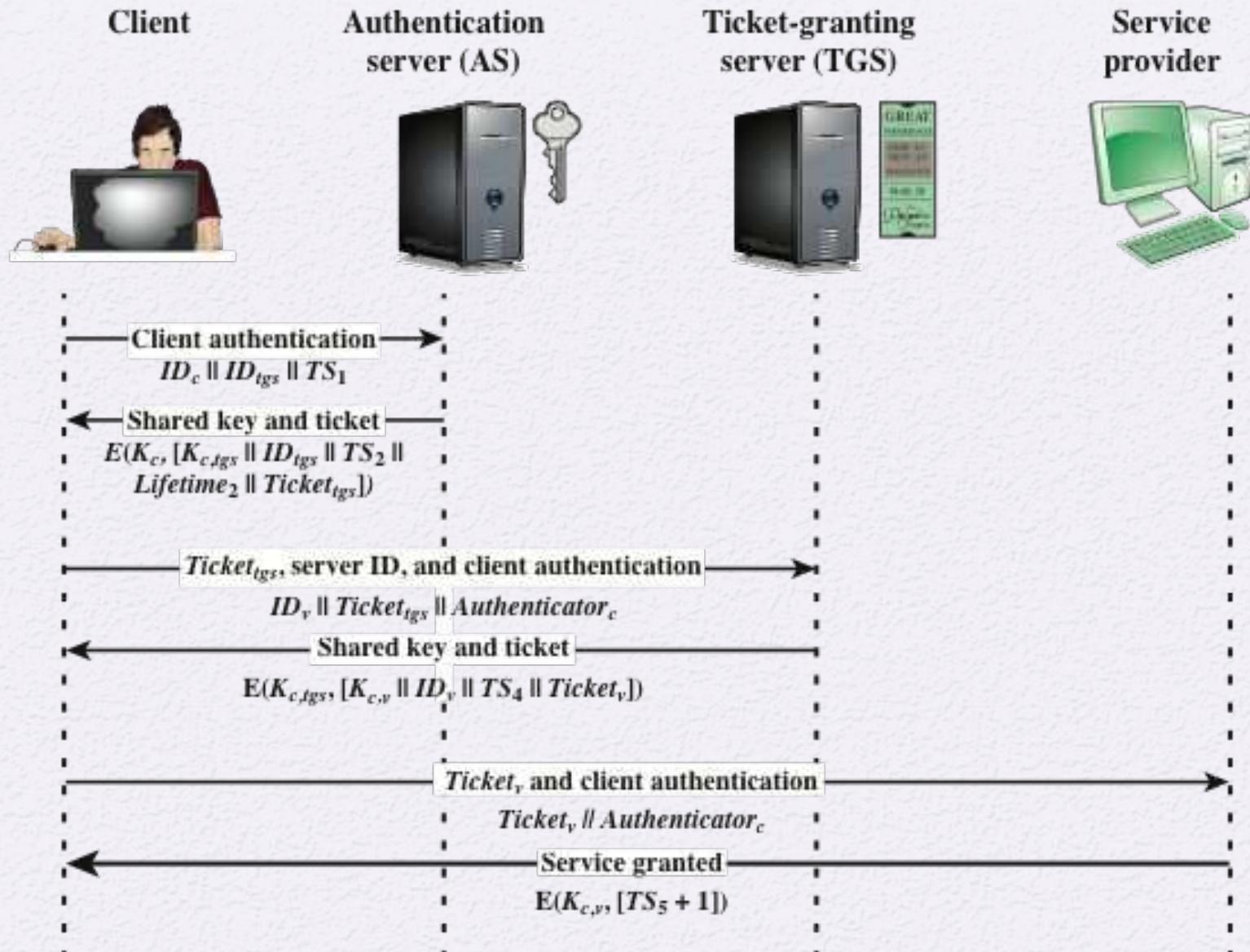


Figure 15.3 Kerberos Exchanges

**Table 15.2 Rationale for the Elements of the Kerberos Version 4 Protocol**  
(page 1 of 3)

<b>Message (1)</b>	Client requests ticket-granting ticket.
$ID_c$	Tells AS identity of user from this client.
$ID_{tgt}$	Tells AS that user requests access to TGS.
$TS_1$	Allows AS to verify that client's clock is synchronized with that of AS.
<b>Message (2)</b>	AS returns ticket-granting ticket.
$K_c$	Encryption is based on user's password, enabling AS and client to verify password, and protecting contents of message (2).
$K_{c,tgs}$	Copy of session key accessible to client created by AS to permit secure exchange between client and TGS without requiring them to share a permanent key.
$ID_{tgt}$	Confirms that this ticket is for the TGS.
$TS_2$	Informs client of time this ticket was issued.
$Lifetime_2$	Informs client of the lifetime of this ticket.
$Ticket_{tgt}$	Ticket to be used by client to access TGS.

(This table can be found on pages 473 – 474 in the textbook)

<b>Message (3)</b>	Client requests service-granting ticket.
$ID_V$	Tells TGS that user requests access to server V.
$Ticket_{tgs}$	Assures TGS that this user has been authenticated by AS.
$Authenticator_c$	Generated by client to validate ticket .
<b>Message (4)</b>	TGS returns service-granting ticket.
$K_{c,tgs}$	Key shared only by C and TGS protects contents of message (4).
$K_{c,v}$	Copy of session key accessible to client created by TGS to permit secure exchange between client and server without requiring them to share a permanent key.
$ID_V$	Confirms that this ticket is for server V.
$TS_4$	Informs client of time this ticket was issued.
$Ticket_v$	Ticket to be used by client to access server V.
$Ticket_{tgs}$	Reusable so that user does not have to reenter password.
$K_{tgs}$	Ticket is encrypted with key known only to AS and TGS, to prevent Tampering.
$K_{c,tgs}$	Copy of session key accessible to TGS used to decrypt authenticator, thereby authenticating ticket.
$ID_C$	Indicates the rightful owner of this ticket.
$AD_C$	Prevents use of ticket from workstation other than one that initially requested the ticket.
$ID_{tgs}$	Assures server that it has decrypted ticket properly.
$TS_2$	Informs TGS of time this ticket was issued.
$Lifetime_2$	Prevents replay after ticket has expired.
$Authenticator_c$	Assures TGS that the ticket presenter is the same as the client for whom the ticket was issued has very short lifetime to prevent replay.
$K_{c,tgs}$	Authenticator is encrypted with key known only to client and TGS, to prevent tampering.
$ID_C$	Must match ID in ticket to authenticate ticket.
$AD_C$	Must match address in ticket to authenticate ticket.
$TS_3$	Informs TGS of time this authenticator was generated.

<b>Message (5)</b>	Client requests service.
$Ticket_V$	Assures server that this user has been authenticated by AS.
$Authenticator_c$	Generated by client to validate ticket.
<b>Message (6)</b>	Optional authentication of server to client.
$K_{c,s}$	Assures C that this message is from V.
$TS_5 + 1$	Assures C that this is not a replay of an old reply.
$Ticket_v$	Reusable so that client does not need to request a new ticket from TGS for each access to the same server.
$K_v$	Ticket is encrypted with key known only to TGS and server, to prevent Tampering.
$K_{c,v}$	Copy of session key accessible to client; used to decrypt authenticator, thereby authenticating ticket.
$ID_C$	Indicates the rightful owner of this ticket.
$AD_C$	Prevents use of ticket from workstation other than one that initially requested the ticket.
$ID_V$	Assures server that it has decrypted ticket properly.
$TS_4$	Informs server of time this ticket was issued.
$Lifetime_4$	Prevents replay after ticket has expired.
$Authenticator_c$	Assures server that the ticket presenter is the same as the client for whom the ticket was issued; has very short lifetime to prevent replay.
$K_{c,v}$	Authenticator is encrypted with key known only to client and server, to prevent tampering.
$ID_C$	Must match ID in ticket to authenticate ticket.
$AD_c$	Must match address in ticket to authenticate ticket.
$TS_5$	Informs server of time this authenticator was generated.

# Kerberos Realms and Multiple Kerberi

- A full-service Kerberos environment consisting of a Kerberos server, a number of clients, and a number of application servers requires that:
  - The Kerberos server must have the user ID and hashed passwords of all participating users in its database; all users are registered with the Kerberos server
  - The Kerberos server must share a secret key with each server; all servers are registered with the Kerberos server
  - The Kerberos server in each interoperating realm shares a secret key with the server in the other realm; the two Kerberos servers are registered with each other

# Kerberos Realm

- A set of managed nodes that share the same Kerberos database
- The database resides on the Kerberos master computer system, which should be kept in a physically secure room
- A read-only copy of the Kerberos database might also reside on other Kerberos computer systems
- All changes to the database must be made on the master computer system
- Changing or accessing the contents of a Kerberos database requires the Kerberos master password

# Kerberos Principal

- A service or user that is known to the Kerberos system
- Identified by its principal name
  - A service or user name
  - An instance name
  - A realm name
  - Three parts of a principal name

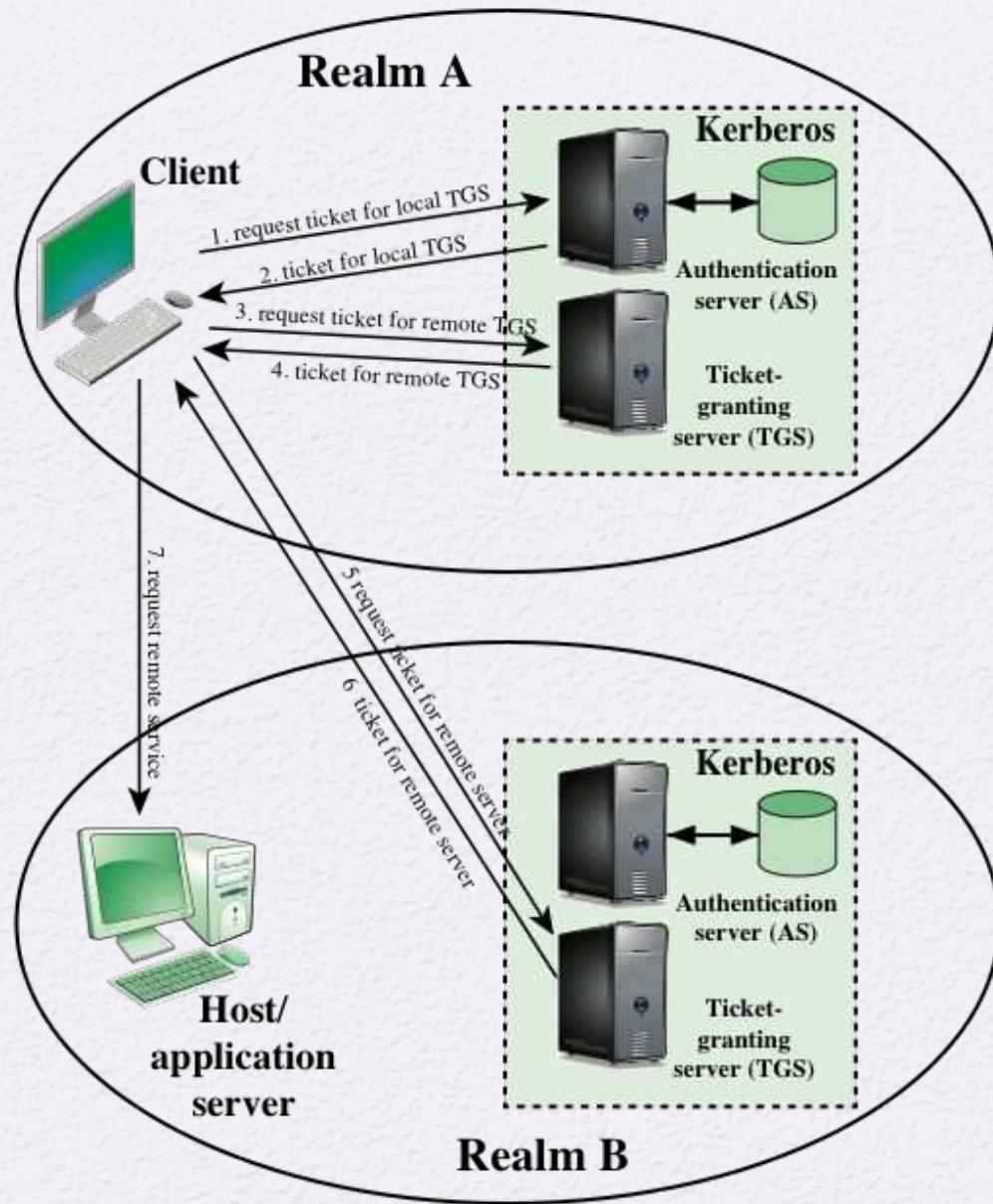


Figure 15.4 Request for Service in Another Realm

# Differences Between Versions 4 and 5

Table 15.3 (page 479 in textbook)

# Summary of Kerberos Version 5 Message Exchanges

(1)  $C \rightarrow AS$   $Options \parallel ID_c \parallel Realm_c \parallel ID_{TGS} \parallel Times \parallel Nonce_1$

(2)  $AS \rightarrow C$   $Realm_c \parallel IDC \parallel Ticket_{TGS} \parallel E(K_{c,TGS}, [K_{c,TGS} \parallel Times \parallel Nonce_1 \parallel Realm_{TGS} \parallel ID_{TGS}])$

$Ticket_{TGS} = E(K_{TGS}, [Flags \parallel K_{c,TGS} \parallel Realm_c \parallel IDC \parallel ADC \parallel Times])$

(a) Authentication Service Exchange to obtain ticket-granting ticket

(3)  $C \rightarrow TGS$   $Options \parallel ID_v \parallel Times \parallel Nonce_2 \parallel Ticket_{TGS} \parallel Authenticator_c$

(4)  $TGS \rightarrow C$   $Realm_c \parallel IDC \parallel Ticket_v \parallel E(K_{v,TGS}, [K_{v,TGS} \parallel Times \parallel Nonce_2 \parallel Realm_v \parallel ID_v])$

$Ticket_{TGS} = E(K_{TGS}, [Flags \parallel K_{c,TGS} \parallel Realm_c \parallel IDC \parallel ADC \parallel Times])$

$Ticket_v = E(K_v, [Flags \parallel K_{v,TGS} \parallel Realm_c \parallel IDC \parallel ADC \parallel Times])$

$Authenticator_c = E(K_{c,TGS}, [IDC \parallel Realm_c \parallel TSI])$

(b) Ticket-Granting Service Exchange to obtain service-granting ticket

(5)  $C \rightarrow V$   $Options \parallel Ticket_v \parallel Authenticator_c$

(6)  $V \rightarrow C$   $E_{K_{c,V}}[TS_2 \parallel Subkey \parallel Seq\#]$

$Ticket_v = E(K_v, [Flags \parallel K_{v,TGS} \parallel Realm_c \parallel IDC \parallel ADC \parallel Times])$

$Authenticator_c = E(K_{c,V}, [IDC \parallel Realm_c \parallel TS_2 \parallel Subkey \parallel Seq\#])$

(c) Client/Server Authentication Exchange to obtain service

## Table 15.4

# Kerberos Version 5 Flags

INITIAL	This ticket was issued using the AS protocol and not issued based on a ticket-granting ticket.
PRE-AUTHENT	During initial authentication, the client was authenticated by the KDC before a ticket was issued.
HW-AUTHENT	The protocol employed for initial authentication required the use of hardware expected to be possessed solely by the named client.
RENEWABLE	Tells TGS that this ticket can be used to obtain a replacement ticket that expires at a later date.
MAY-POSTDATE	Tells TGS that a postdated ticket may be issued based on this ticket-granting ticket.
POSTDATED	Indicates that this ticket has been postdated; the end server can check the authtime field to see when the original authentication occurred.
INVALID	This ticket is invalid and must be validated by the KDC before use.
PROXiable	Tells TGS that a new service-granting ticket with a different network address may be issued based on the presented ticket.
PROXY	Indicates that this ticket is a proxy.
FORWARDABLE	Tells TGS that a new ticket-granting ticket with a different network address may be issued based on this ticket-granting ticket.
FORWARDED	Indicates that this ticket has either been forwarded or was issued based on authentication involving a forwarded ticket-granting ticket.

(Table can be found on page 480 in textbook)

# Mutual Authentication

- Public-key encryption for session key distribution
  - Assumes each of the two parties is in possession of the current public key of the other
  - May not be practical to require this assumption
- Denning protocol using timestamps
  - Uses an authentication server (AS) to provide public-key certificates
  - Requires the synchronization of clocks
- Woo and Lam makes use of nonces
  - Care needed to ensure no protocol flaws

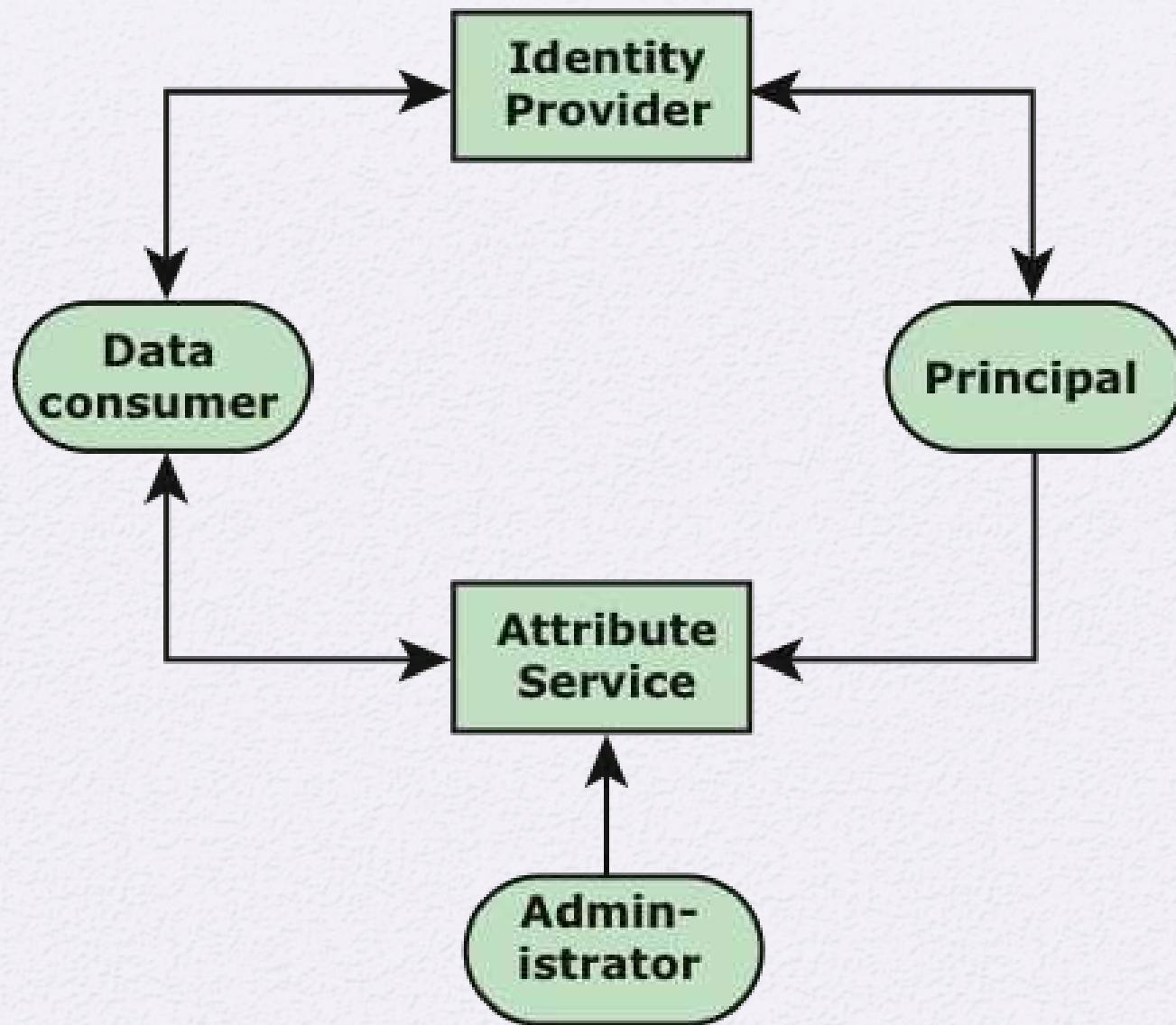
# One-Way Authentication

- Have public-key approaches for e-mail
  - Encryption of message for confidentiality, authentication, or both
  - The public-key algorithm must be applied once or twice to what may be a long message
- For confidentiality encrypt message with one-time secret key, public-key encrypted
- If authentication is the primary concern, a digital signature may suffice

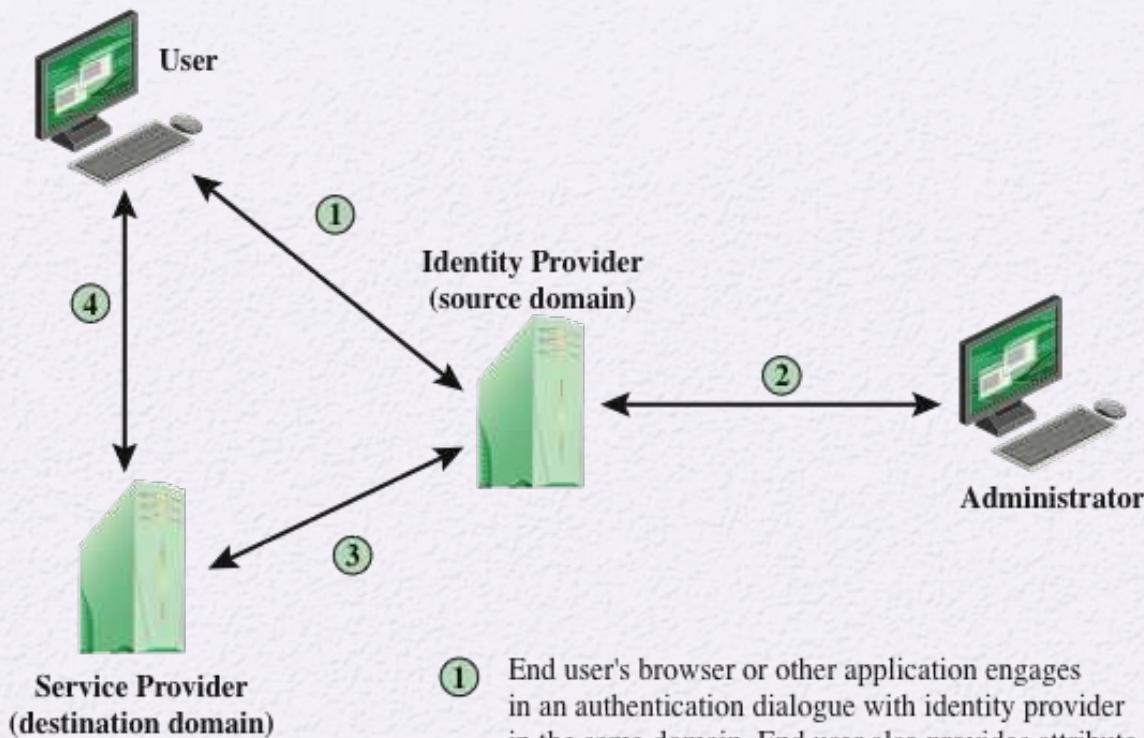
# Federated Identity Management

- Relatively new concept dealing with the use of a common identity management scheme across multiple enterprise and numerous applications and supporting many users
- Services provided include:
  - Point of contact
  - SSO protocol services
  - Trust services
  - Key services
  - Identity services
  - Authorization
  - Provisioning
  - Management





**Figure 15.5 Generic Identity Management System**

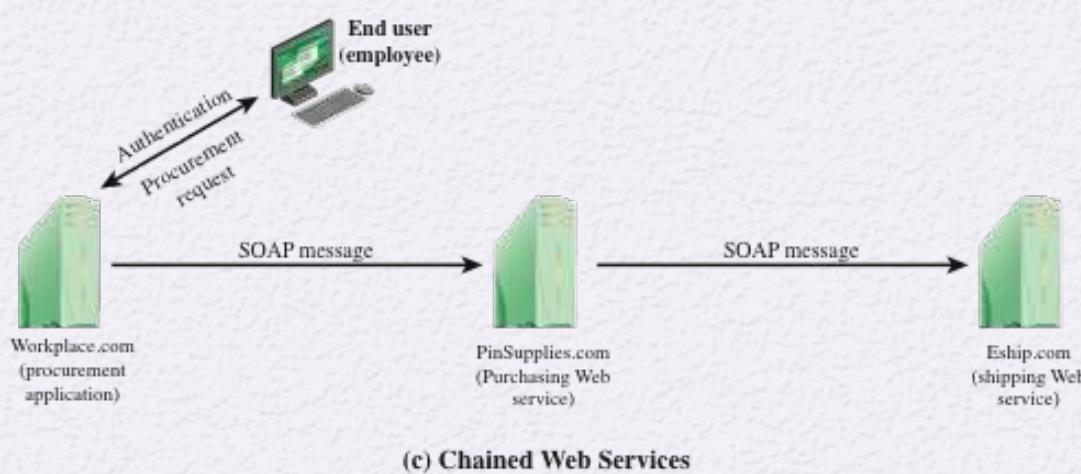
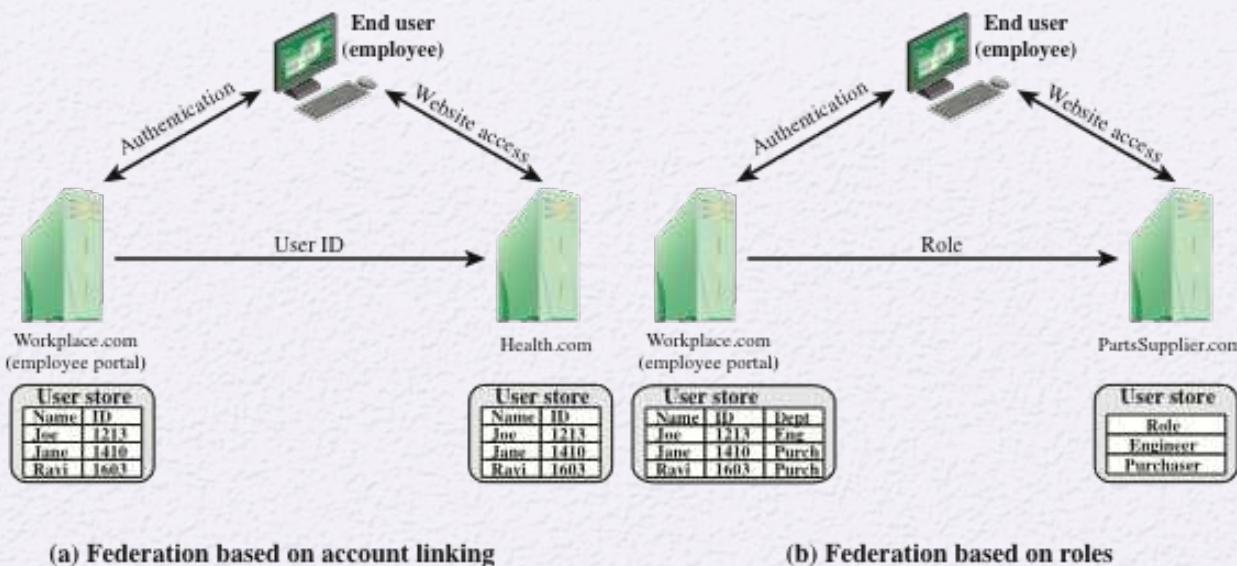


- ① End user's browser or other application engages in an authentication dialogue with identity provider in the same domain. End user also provides attribute values associated with user's identity.
- ② Some attributes associated with an identity, such as allowable roles, may be provided by an administrator in the same domain.
- ③ A service provider in a remote domain, which the user wishes to access, obtains identity information, authentication information, and associated attributes from the identity provider in the source domain.
- ④ Service provider opens session with remote user and enforces access control restrictions based on user's identity and attributes.

**Figure 15.6 Federated Identity Operation**

# Key Standards

- **The Extensible Markup Language (XML)**
  - A markup language that uses sets of embedded tags or labels to characterize text elements within a document so as to indicate their appearance, function, meaning, or context
- **The Simple Object Access Protocol (SOAP)**
  - Enables applications to request services from one another with XML-based requests and receive responses as data formatted with XML
- **WS-Security**
  - A set of SOAP extensions for implementing message integrity and confidentiality in Web services
- **Security Assertion Markup Language (SAML)**
  - An XML-based language for the exchange of security information between online business partners



**Figure 15.7 Federated Identity Scenarios**

# Personal Identity Verification

- User authentication based on the possession of a smart card is becoming more widespread
  - Has the appearance of a credit card
  - Has an electronic interface
  - May use a variety of authentication protocols
- A smart card contains within it an entire microprocessor, including processor, memory, and I/O ports
- A smart card includes three types of memory:
  - Read-only memory (ROM) stores data that does not change during the card's life
  - Electronically erasable programmable ROM (EEPROM) holds application data and programs; also holds data that may vary with time
  - Random access memory (RAM) holds temporary data generated when applications are executed

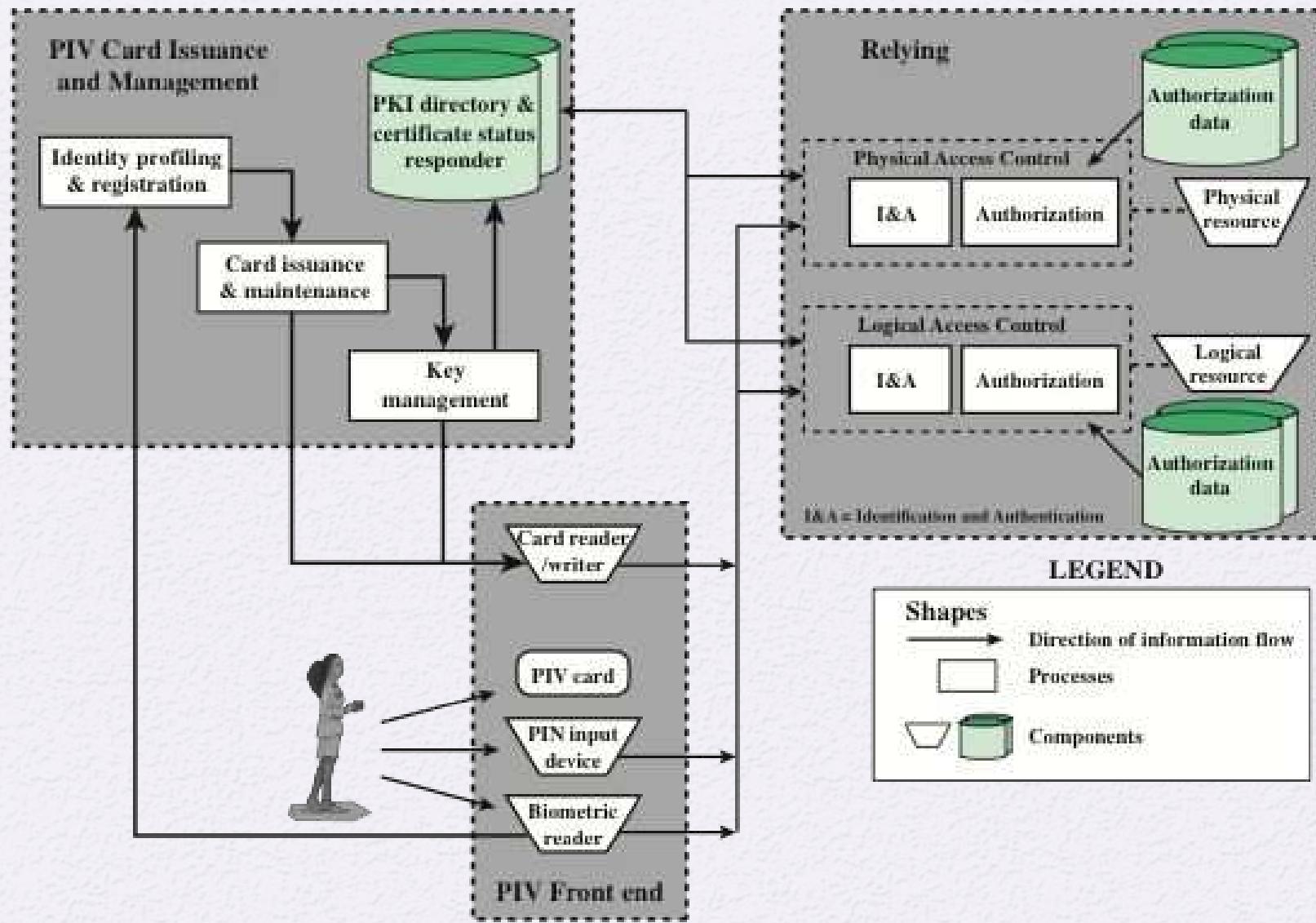


Figure 15.8 FIPS 201 PIV System Model

# PIV Documentation

- **FIPS 201-2—Personal Identity Verification (PIV) of Federal Employees and Contractors**
  - Specifies the physical card characteristics, storage media, and data elements that make up the identity credentials resident on the PIV card
- **SP 800-73-3—Interfaces for Personal Identity Verification**
  - Specifies the interfaces and card architecture for storing and retrieving identity credentials from a smart card, and provides guidelines for the use of authentication mechanisms and protocols
- **SP 800-76-2—Biometric Data Specification for Personal Identity Verification**
  - Describes technical acquisition and formatting specifications for the biometric credentials of the PIV system
- **SP 800-78-3—Cryptographic Algorithms and Key Sizes for Personal Identity Verification**
  - Identifies acceptable symmetric and asymmetric encryption algorithms, digital signature algorithms, and message digest algorithms, and specifies mechanisms to identify the algorithms associated with PIV keys or digital signatures
- **SP 800-104—A Scheme for PIV Visual Card Topography**
  - Provides additional recommendations on the PIV card color-coding for designating employee affiliation
- **SP 800-116—A Recommendation for the Use of PIV Credentials in Physical Access Control Systems (PACS)**
  - Describes a risk-based approach for selecting appropriate PIV authentication mechanisms to manage physical access to Federal government facilities and assets
- **SP 800-79-1—Guidelines for the Accreditation of Personal Identity Verification Card Issuers**
  - Provides guidelines for accrediting the reliability of issuers of PIV cards that collect, store, and disseminate personal identity credentials and issue smart cards
- **SP 800-96—PIV Card to Reader Interoperability Guidelines**
  - Provides requirements that facilitate interoperability between any card and any reader

# PIV Credentials and Keys

- **Personal Identification Number (PIN)**
  - Required to activate the card for privileged operation
- **Cardholder Unique Identifier (CHUID)**
  - Includes the Federal Agency Smart Credential Number (FASC-N) and the Global Unique Identification Number (GUID), which uniquely identify the card and the cardholder
- **PIV Authentication Key**
  - Asymmetric key pair and corresponding certificate for user authentication
- **Two fingerprint templates**
  - For biometric authentication
- **Electronic facial image**
  - For biometric authentication
- **Asymmetric Card Authentication Key**
  - Asymmetric key pair and corresponding certificate used for card authentication

## Optional elements include the following:

- **Digital Signature Key**
  - Asymmetric key pair and corresponding certificate that supports document signing and signing of data elements such as the CHUID
- **Key Management Key**
  - Asymmetric key pair and corresponding certificate supporting key establishment and transport
- **Symmetric Card Authentication Key**
  - For supporting physical access applications
- **PIV Card Application Administration Key**
  - Symmetric key associated with the card management system
- **One or two iris images**
  - For biometric authentication

**Table 15.5**  
**PIV Algorithms and Key Sizes**

<b>PIV Key Type</b>	<b>Algorithms</b>	<b>Key Sizes (bits)</b>	<b>Application</b>
PIV Authentication Key	RSA	2048	Supports card and cardholder authentication for an interoperable environment.
	ECDSA	256	
Card Authentication Key	3TDEA	168	Supports card authentication for physical access
	AES	128, 192, or 256	
	RSA	2048	Supports card authentication for an interoperable environment.
	ECDSA	256	
Digital Signature Key	RSA	2048 or 3072	Supports document signing and nonce signing
	ECDSA	256 or 384	
Key Management Key	RSA	2048	Supports key establishment and transport.
	ECDH	256 or 384	

# Authentication

- **Using the electronic credentials resident on a PIV card, the card supports the following authentication mechanisms:**

- **CHUID**

The cardholder is authenticated using the signed CHUID data element on the card.

The PIN is not required. This mechanism is useful in environments where a low level of assurance is acceptable and rapid contactless authentication is necessary

- **Card Authentication Key**

The PIV card is authenticated using the Card Authentication Key in a challenge response protocol. The PIN is not required. This mechanism allows contact (via card reader) or contactless (via radio waves) authentication of the PIV card without the holder's active participation, and provides a low level of assurance

- **BIO**

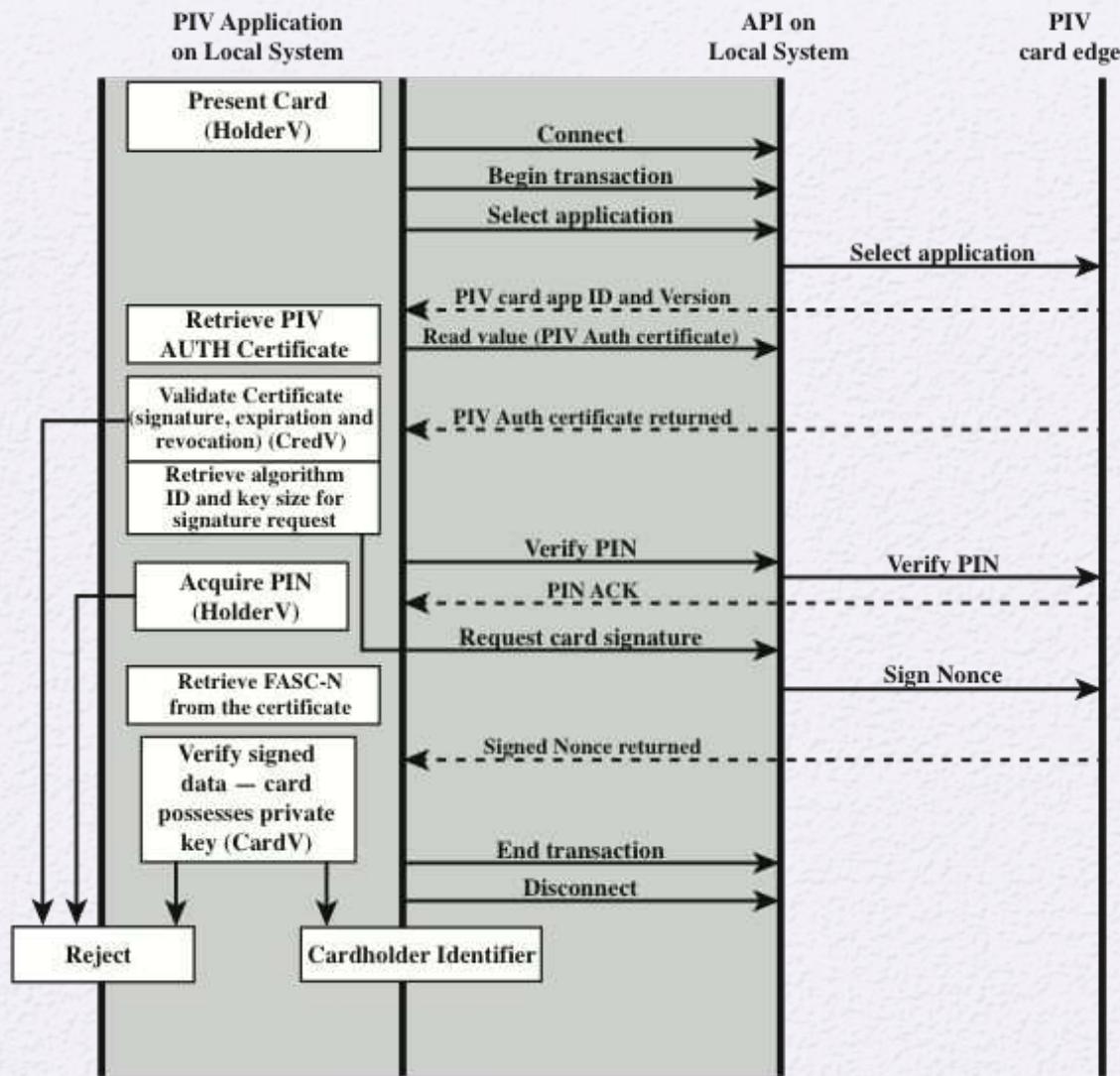
The cardholder is authenticated by matching his or her fingerprint sample(s) to the signed biometric data element in an environment without a human attendant in view. The PIN is required to activate the card. This mechanism achieves a high level of assurance and requires the cardholder's active participation in submitting the PIN as well as the biometric sample

- **BIO-A**

The cardholder is authenticated by matching his or her fingerprint sample(s) to the signed biometric data element in an environment with a human attendant in view. The PIN is required to activate the card. This mechanism achieves a very high level of assurance when coupled with full trust validation of the biometric template retrieved from the card, and requires the cardholder's active participation in submitting the PIN as well as the biometric sample

- **PKI**

The cardholder is authenticated by demonstrating control of the PIV authentication private key in a challenge response protocol that can be validated using the PIV authentication certificate. The PIN is required to activate the card. This mechanism achieves a very high level of identity assurance and requires the cardholder's knowledge of the PIN



CardV = Card validation

CredV = Credential validation

HolderV = Cardholder validation

FASC-N = Federal Agency Smart Credential Number

**Figure 15.9 Authentication using PIV Authentication Key**

# Summary

- Remote user-authentication principles
  - The NIST model for electronic user authentication
  - Means of authentication
  - Mutual authentication
  - One-way authentication
- Remote user-authentication using symmetric encryption
  - Mutual authentication
  - One-way authentication
- Kerberos
  - Motivation
  - Kerberos V4 and V5
- Remote user-authentication using asymmetric encryption
  - Mutual authentication
  - One-way authentication
- Federated identity management
  - Identity management
  - Identity federation
- Personal identity verification
  - PIV system model
  - PIV documentation
  - PIV credentials and keys
  - Authentication



# **ELLIPTIC CURVE CRYPTOGRAPHY**

# Elliptic Curve Cryptography: Motivation

- Public key cryptographic algorithms (asymmetric key algorithms) play an important role in providing security services:
  - Key management
  - Confidentiality
  - User authentication
  - Signature
- Public key cryptography systems are constructed by relying on the **hardness of mathematical problems**
  - RSA: based on the integer factorization problem
  - DH: based on the discrete logarithm problem
- The main problem of conventional public key cryptography systems
  - **key size has to be sufficient large** in order to meet **the high-level security requirement.**
- This results in lower speed and consumption of more bandwidth
  - Solution: **Elliptic Curve Cryptography system**

# Introduction to Elliptic Curves

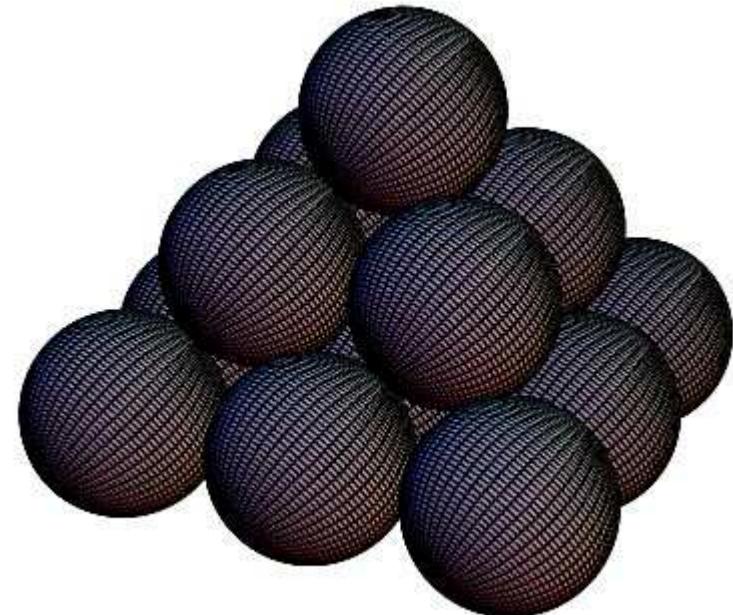
- Lets start with a puzzle...

*What is the number of balls that may be piled as a square pyramid and also rearranged into a square array?*

# Introduction to Elliptic Curves...

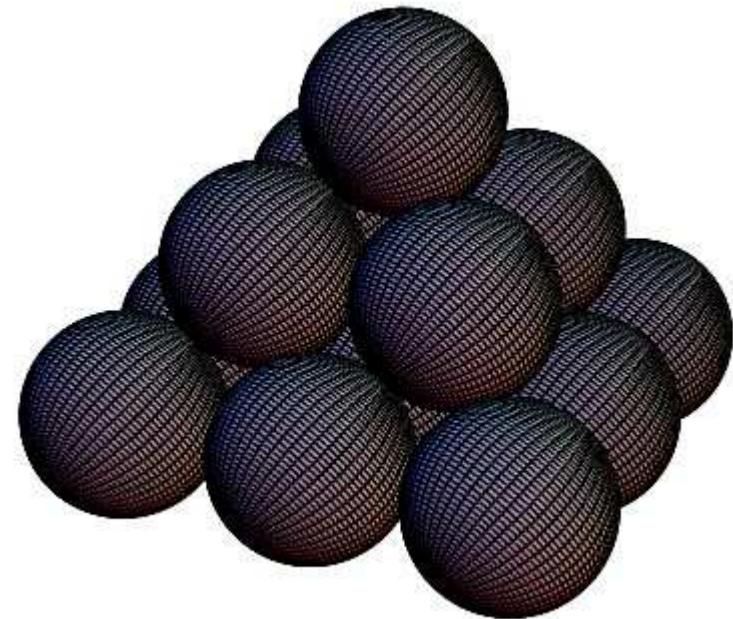
- Lets start with a puzzle...

*What is the number of balls  
that may be piled as a square  
pyramid and also rearranged  
into a square array?*



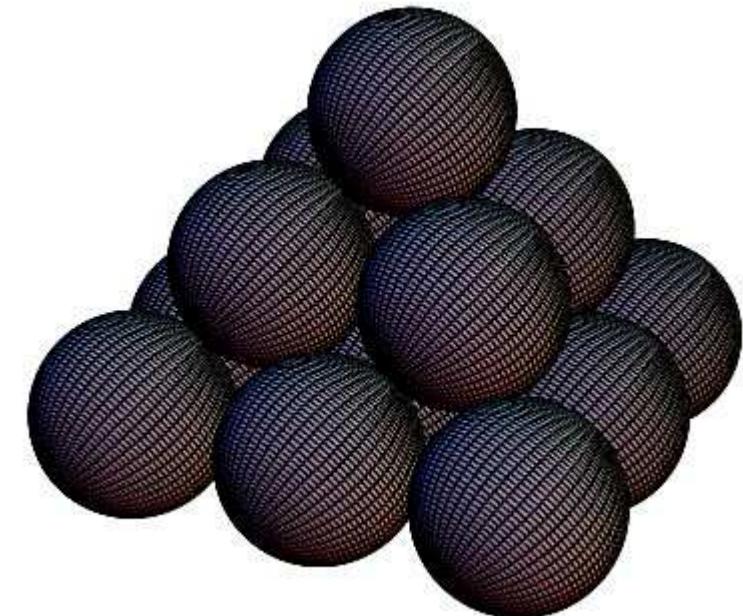
# Introduction to Elliptic Curves...

- What about the figure shown?
- *Does it fulfil our requirements?*



# Introduction to Elliptic Curves...

- What about the figure shown?
- *Does it fulfil our requirements???*
- *Can you find solutions to this problem???*



# Introduction to Elliptic Curves...

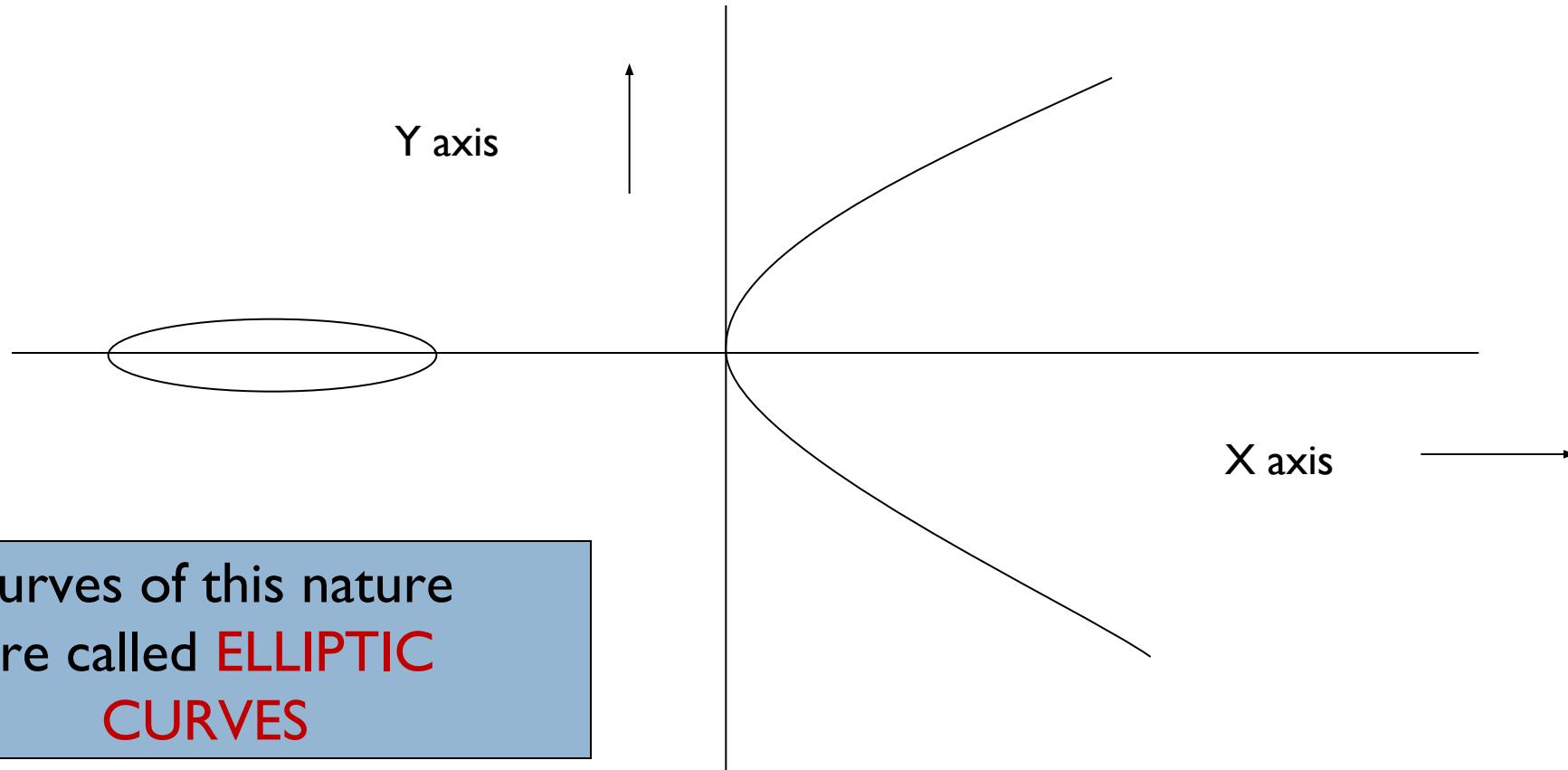
- Let  $x$  be the height of the pyramid, then the number of balls in pyramid is,

$$1^2 + 2^2 + 3^2 + \dots + x^2 = \frac{x(x+1)(2x+1)}{6}$$

- We also want this to be a square. Hence,

$$y^2 = \frac{x(x+1)(2x+1)}{6}$$

# Graphical Representation



# Method of Diophantus

- Uses a set of known points to produce new points
- (0,0) and (1,1) are two trivial solutions
- Equation of line through these points is  $y=x$ .
- Intersecting with the curve and rearranging terms:

$$x^3 - \frac{3}{2}x^2 + \frac{1}{2}x = 0$$

- What are the roots of this equation???

# Method of Diophantus...

- Uses a set of known points to produce new points
- (0,0) and (1,1) are two trivial solutions
- Equation of line through these points is  $y=x$ .
- Intersecting with the curve and rearranging terms:

$$x^3 - \frac{3}{2}x^2 + \frac{1}{2}x = 0$$

- What are the roots of this equation???
- Two trivial roots  $x=0$  and  $x=1$ ..... But what about third one????

# Method of Diophantus...

- We know that, for any numbers  $a, b, c$ , we have,

$$(x-a)(x-b)(x-c) = x^3 - (a+b+c)x^2 + (ab+bc+ac)x - abc$$

- Hence, for the equation

$$x^3 - \frac{3}{2}x^2 + \frac{1}{2}x = 0$$

- We have,

$$a + b + \frac{x}{2} = \quad \square 0 + 1 + x \frac{\frac{3}{2}}{2} \quad \square x \frac{\pm}{2}$$

- Hence, one more point  $(\frac{1}{2}, \frac{1}{2})$  and because of the symmetry , another  $(\frac{1}{2}, -\frac{1}{2})$

## Method of Diophantus... : Exercise

- Can you find out another point on curve using Diophantus's method ???

*Consider two points  $(\frac{1}{2}, -\frac{1}{2})$  and  $(1, 1)$  and find out another point on the curve .....*

## Method of Diophantus... : Exercise solution

- Consider the line through  $(1/2, -1/2)$  and  $(1, 1) \Rightarrow y=3x-2$
- Intersecting with the curve we have:

$$x^3 - \frac{51}{2}x^2 + \dots = 0$$

- Thus  $\frac{1}{2} + 1 + x = 51/2$  or  $x = 24$  and  $y=70$
- Thus if we have 4900 balls we may arrange them in either way

# Weierstrass Equation

- For most situations, an elliptic curve  $E$  is the graph of an equation of the form:

$$y^2 = x^3 + Ax + B$$

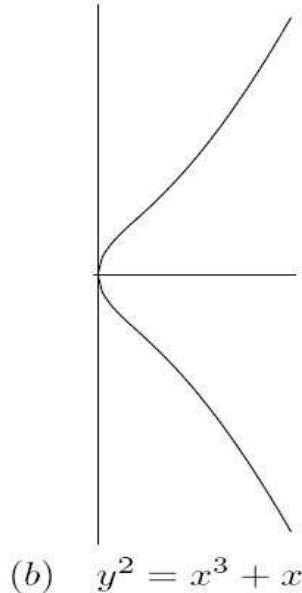
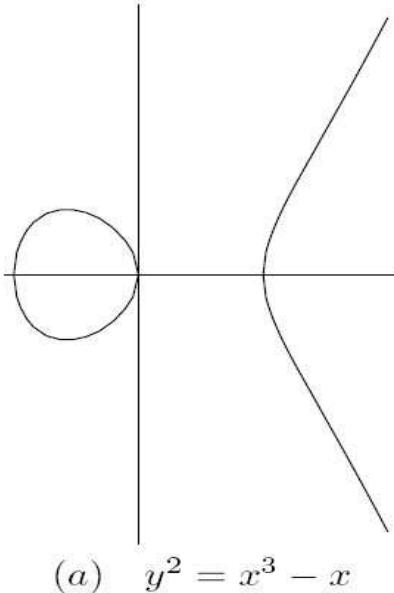
where  $A$  and  $B$  are constants. This refers to the Weierstrass Equation of Elliptic Curve.

- Here,  $A$ ,  $B$ ,  $x$  and  $y$  all belong to a field of say rational numbers, complex numbers, finite fields ( $F_p$ ) or Galois Fields ( $GF(2^n)$ ).
- If  $K$  is the field where  $A, B \in K$ , then we say that the **Elliptic Curve  $E$  is defined over  $K$**

# Points on Elliptic Curve

- If we want to consider points with coordinates in some field  $L$ , we write  $E(L)$ . By definition, this set always contains the point at infinity  $O$

$$E(L) = \{O\} \cup \{(x, y) \in L \times L \mid y^2 = x^3 + Ax + B\}$$

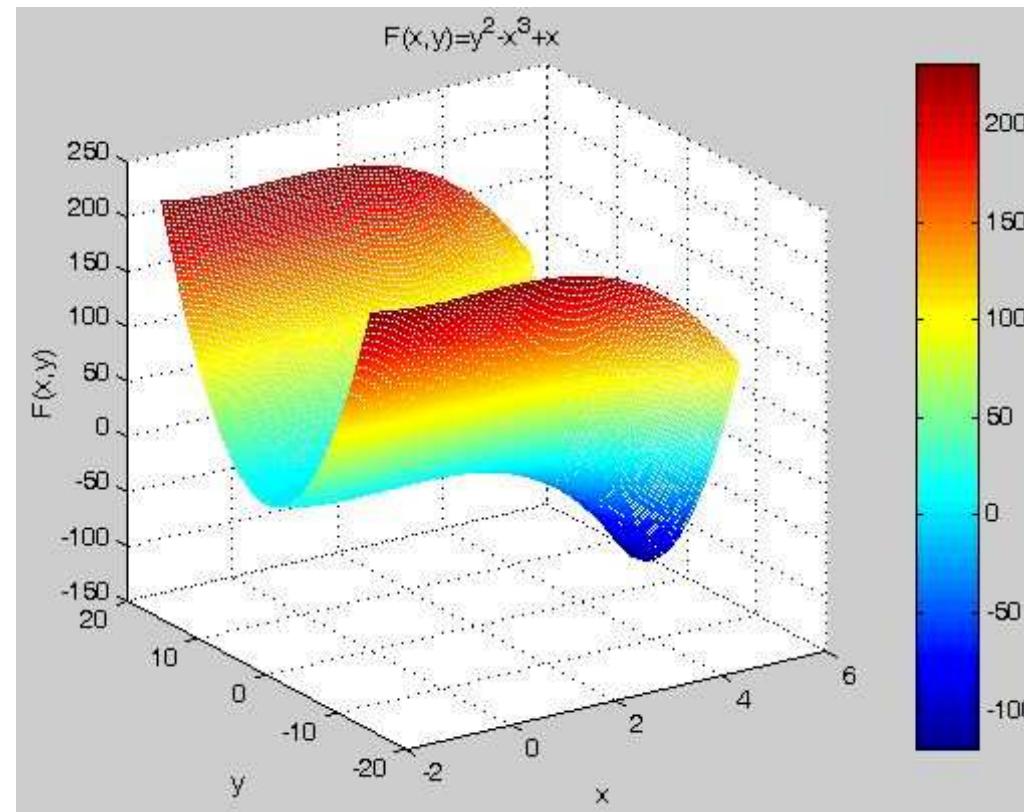


What about the roots of these curves ????

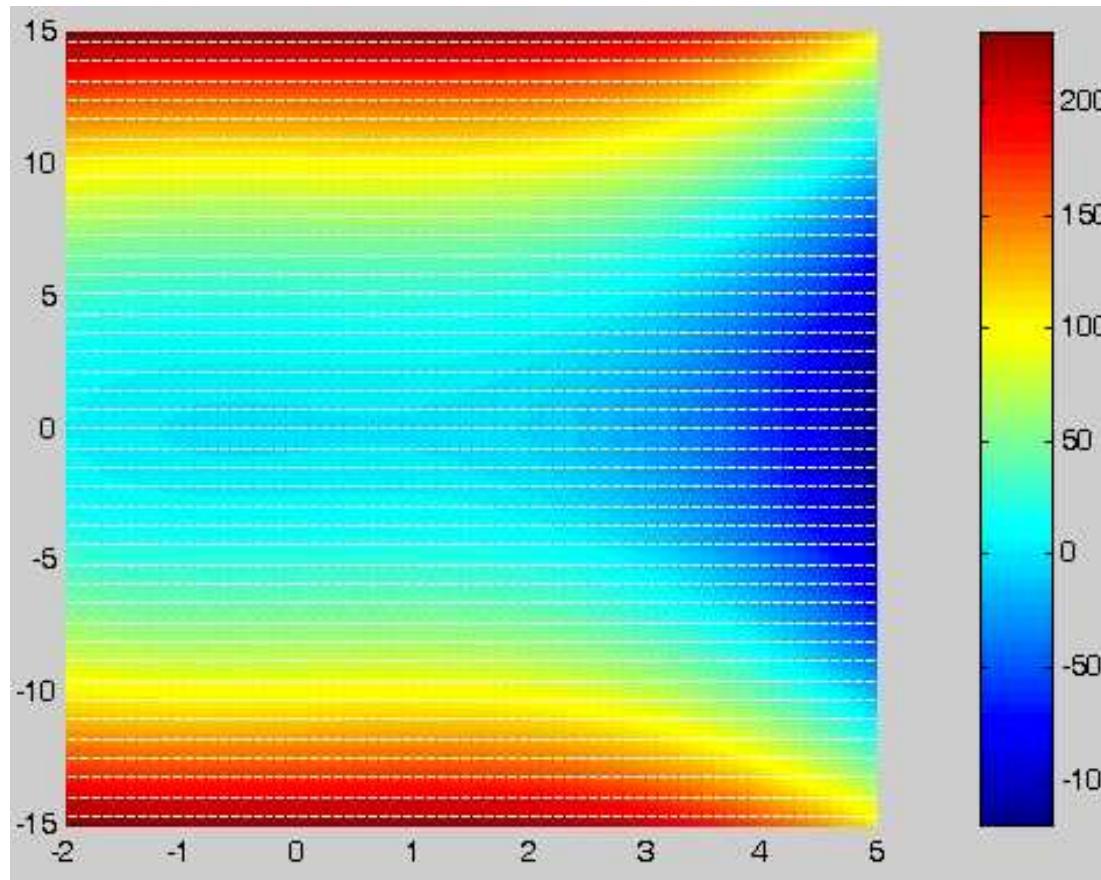
We must have the equation  
 $4A^3 + 27B^2 \neq 0$  satisfied

A condition for an Elliptic curve to be a group !!!!

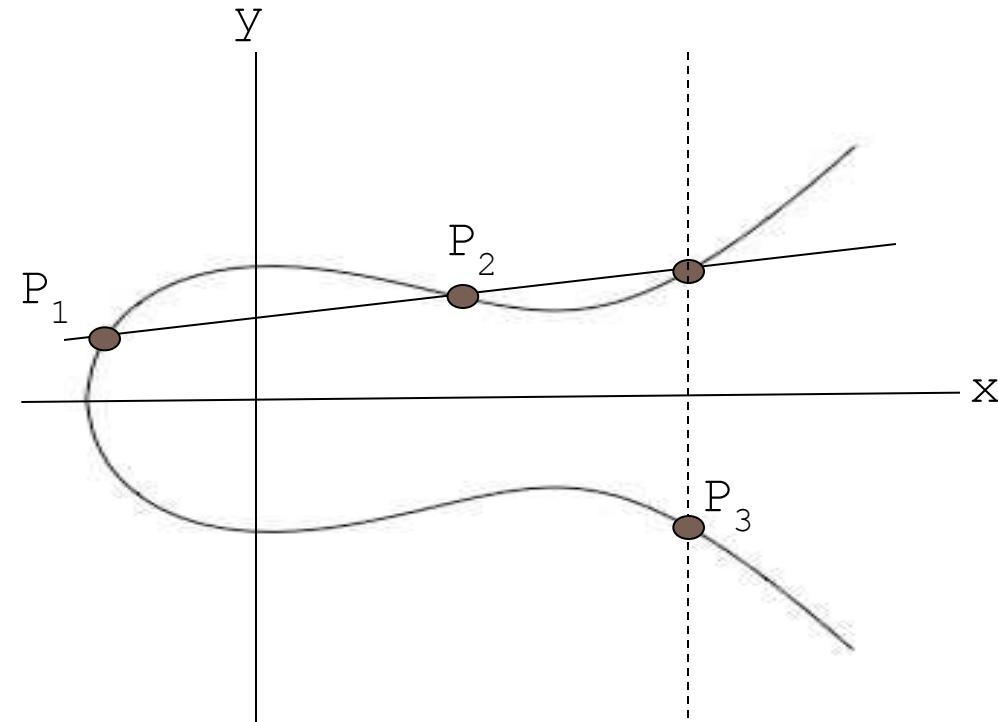
# Points on Elliptic Curve...



# Points on Elliptic Curve...



# Adding points on Elliptic Curve...



- Consider elliptic curve  
 $E: y^2 = x^3 - x + 1$
- Start with two points :  $P_1(x_1,y_1)$  and  $P_2(x_2,y_2)$  on elliptic curve
- To get a new point  $P_3$ ,
  - Draw a line L through  $P_1$  and  $P_2$
  - Get the intersection  $P'_3$
  - Reflect across x-axis to get  $P_3$
- We define  $P_1 + P_2 = P_3$

# Adding points on Elliptic Curve...

- Case 1:  $P_1 \neq P_2$  and neither point is O
  - For  $x_1 \neq x_2$
  - For  $x_1 = x_2$  ????
  - We get  $P_1 + P_2 = O$

Slope of the line L passing through  $P_1$  and  $P_2$  is,

$$m = \frac{(y_2 - y_1)}{(x_2 - x_1)}$$

For  $x_1 \neq x_2$ , equation of line L is,

$$y = m(x - x_1) + y_1$$

To find intersection with E, substitute to get,

$$(m(x - x_1) + y_1)^2 = x^3 + Ax + B$$

Rearrange to form,

$$0 = x^3 - m^2 x^2 + \dots$$

Given two roots  $x_1$  and  $x_2$ , third root can be calculated,

$$(a + b + c) = m^2 \Rightarrow (x_1 + x_2 + x) = m^2$$

$$\Rightarrow x = m^2 - x_1 - x_2$$

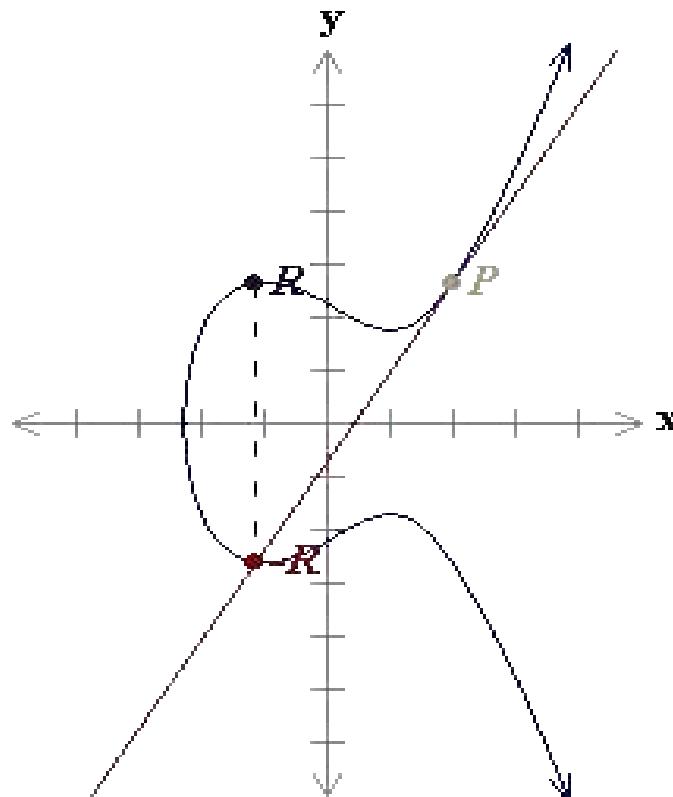
$$\text{and } y = m(x - x_1) + y_1$$

reflecting across the x - axis to obtain the point  $P_3 = (x_3, y_3)$ :

$$x_3 = m^2 - x_1 - x_2 \text{ and } y_3 = m(x_1 - x_3) - y_1$$

# Adding points on Elliptic Curve...

- Case II :  $P_1 = P_2 = (x_1, y_1)$ 
  - When two points on a curve are very close to each other, the line through them approximates a tangent line. Therefore, when the two points coincide, we take the line  $L$  through them to be the tangent line.
  - Implicit differentiation allows us to find the slope  $m$  of  $L$



$$\begin{aligned}P & (2, 2.65) \\-R & (-1.11, -2.64) \\R & (-1.11, 2.64)\end{aligned}$$

$$2P = R = (-1.11, 2.64).$$

$$y^2 = x^3 - 3x + 5$$

# Adding points on Elliptic Curve...

- Case II :  $P_1 = P_2 = (x_1, y_1)$ 
  - When two points on a curve are very close to each other, the line through them approximates a tangent line. Therefore, when the two points coincide, we take the line  $L$  through them to be the tangent line.
  - Implicit differentiation allows us to find the slope  $m$  of  $L$

$$2y \frac{dy}{dx} = 3x^2 + A, \text{ so } m = \frac{dy}{dx} = \frac{3x_1^2 + A}{2y_1}$$

If  $y_1 \neq 0$ , the equation of  $L$  is,

$$y = m(x - x_1) + y_1$$

We find the cubic equation,

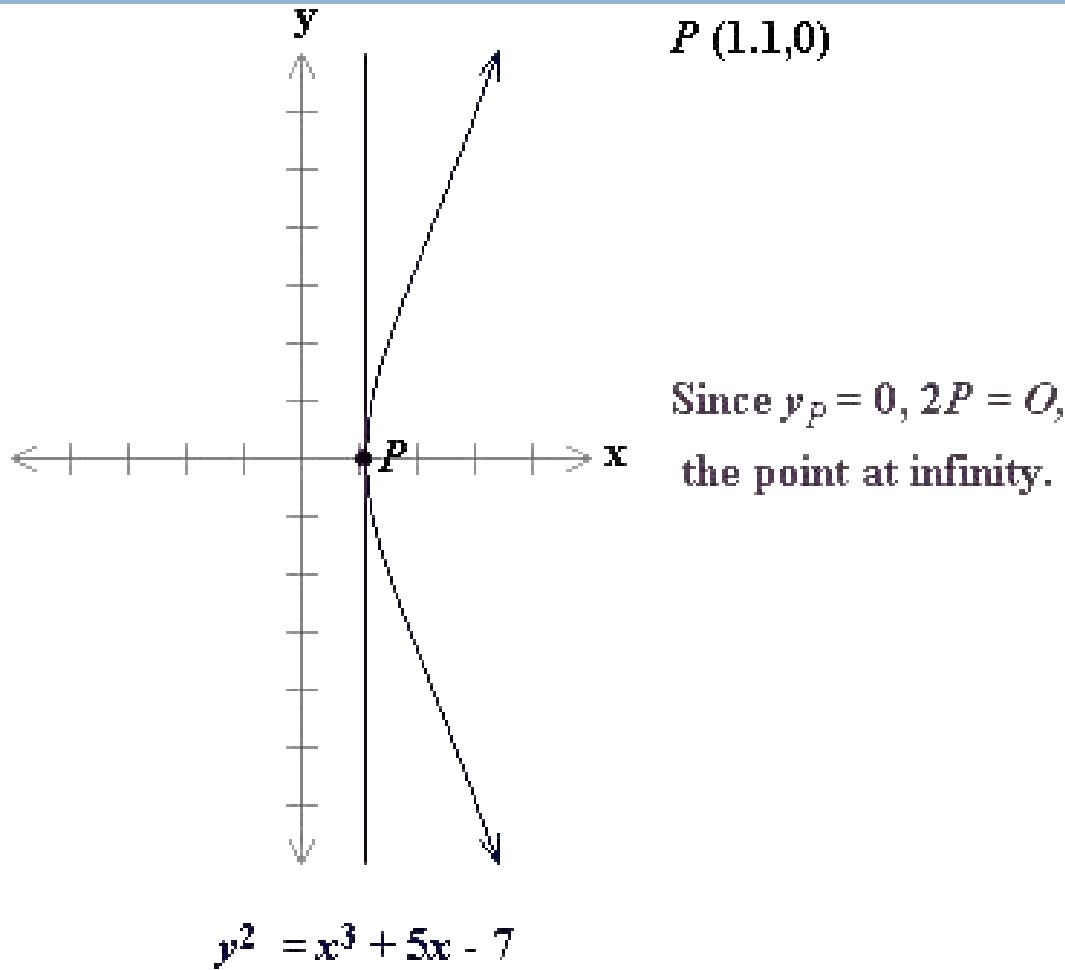
$$0 = x^3 - m^2 x^2 + \dots$$

This time we know only one root  $x_1$ , we obtain :

$$x_3 = m^2 - 2x_1, \quad y_3 = m(x_1 - x_3) - y_1$$

# Adding points on Elliptic Curve...

- Case II :  $P_1 = P_2 = (x_1, y_1)$ 
  - If  $y_1 = 0$ 
    - We get  $P_1 + P_2 = O$
- Case III:  $P_2 = O$ 
  - What about  $P_1 + P_2$  ????
  - Do we get  $P_1 + P_2 = P_1$  ??
  - In other words,  $P_1 + O = P_1$



Since  $y_P = 0$ ,  $2P = O$ ,  
the point at infinity.

$$y^2 = x^3 + 5x - 7$$

# Group Law

- The addition of points on an elliptic curve  $E$  satisfies the following properties:
  - (Commutativity) :  $P_1 + P_2 = P_2 + P_1$  for all  $P_1, P_2$  on  $E$
  - (Existence of identity) :  $P + O = P$  for all  $P$  on  $E$
  - (Existence of inverses) : Given  $P$  on  $E$ , there exists  $P'$  on  $E$  with  $P + P' = O$ . This point  $P'$  will usually be denoted as  $-P$
  - (Associativity) :  $(P_1 + P_2) + P_3 = P_1 + (P_2 + P_3)$  for all  $P_1, P_2, P_3$  on  $E$

The points on  $E$  form an additive abelian group with  $O$  as the identity element.

# Integer times a point

- Let  $k$  be a positive integer and let  $P$  be a point on an elliptic curve, then
  - $kP$  denotes  $P + P + \dots + P$  (with  $k$  summands)
- Efficient computation for large  $k$ 
  - Successive doubling method
    - For example, to compute  $19P$ , we compute
      - $2P, 4P = 2P+2P, 8P = 4P+4P, 16P = 8P+8P, 19P = 16P+2P+P.$
- But, the only difficulty is....
  - The size of the coordinates of the points increases very rapidly if we are working over the rational numbers
  - What about finite fields ????



# **ELLIPTIC CURVES IN CRYPTOGRAPHY**

# Elliptic curves in Cryptography

- Elliptic Curve (EC) systems as applied to cryptography were first proposed in 1985 independently by Neal Koblitz and Victor Miller.
- The **discrete logarithm problem** on elliptic curve groups
  - More difficult than the corresponding problem in (the multiplicative group of nonzero elements of) the underlying **finite field**.

# Why finite field?

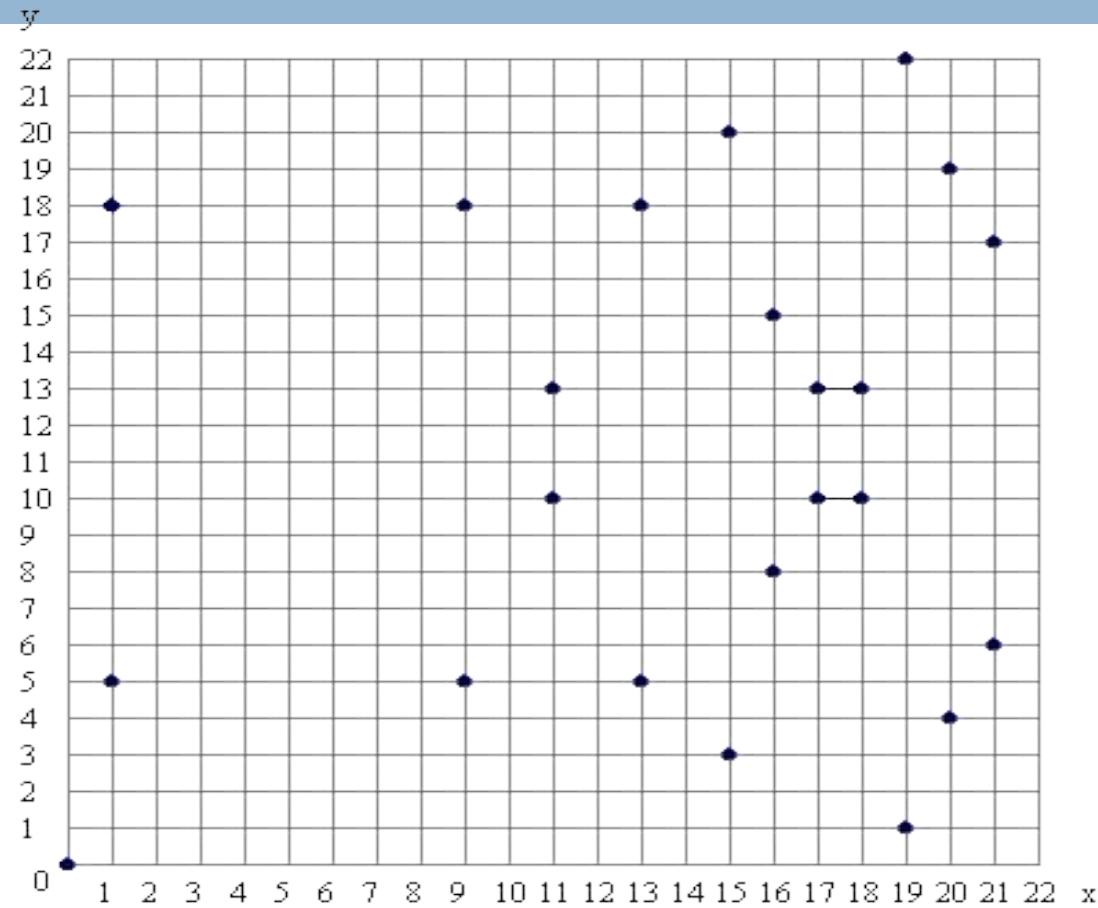
- Elliptic curves over real numbers
  - Calculations prove to be slow
  - Inaccurate due to rounding error
  - Infinite field
- Cryptographic schemes need fast and accurate arithmetic
- In the cryptographic schemes, elliptic curves over two finite fields are mostly used.
  - Prime field  $F_p$ , where p is a prime.
  - Binary field  $F_{2^m}$ , where m is a positive integer

# Elliptic Curve over finite field $F_{23}$

- As a very small example, consider an elliptic curve over the field  $F_{23}$ . With  $a = 1$  and  $b = 0$ , the elliptic curve equation is  $y^2 = x^3 + x$ .
- The point  $(9,5)$  satisfies this equation since:  
 $y^2 \bmod p = x^3 + x \bmod p$   
 $25 \bmod 23 = 729 + 9 \bmod 23$   
 $25 \bmod 23 = 738 \bmod 23$   
 $2 = 2$
- The 23 points which satisfy this equation are:

$(0,0) (1,5) (1,18) (9,5) (9,18) (11,10) (11,13) (13,5) (13,18) (15,3) (15,20) (16,8)$   
 $(16,15) (17,10) (17,13) (18,10) (18,13) (19,1) (19,22) (20,4) (20,19) (21,6) (21,17)$

# Elliptic Curve over finite field $F_{23}$ ...



Elliptic curve equation:  $y^2 = x^3 + x$  over  $F_{23}$

# Elliptic curves over finite fields

- Let us do an exercise....
- Let  $E$  be the curve  $y^2 = x^3 + x + 1$  over  $F_5$ , find all the points on  $E$

Therefore,  $E(F_5)$  has order 9.

Can you show that  $E(F_5)$  is cyclic??? What is the generator??

x	$x^3+x+1$	y	Points
0	1	$\pm 1$	(0,1),(0,4)
1	3	-	-
2	1	$\pm 1$	(2,1),(2,4)
3	1	$\pm 1$	(3,1),(3,4)
4	4	$\pm 2$	(4,2),(4,3)
0		0	O

# Elliptic curves over finite fields... : Exercise

- Let  $E$  be the curve  $y^2 = x^3 + 2$  over  $\mathbb{F}_7$ , find all the points on  $E$

*What is the order of  $E(F_7)$ ?*

*Is  $E(F_{\sqrt{d}})$  cyclic??? If yes,  
what is the generator??*

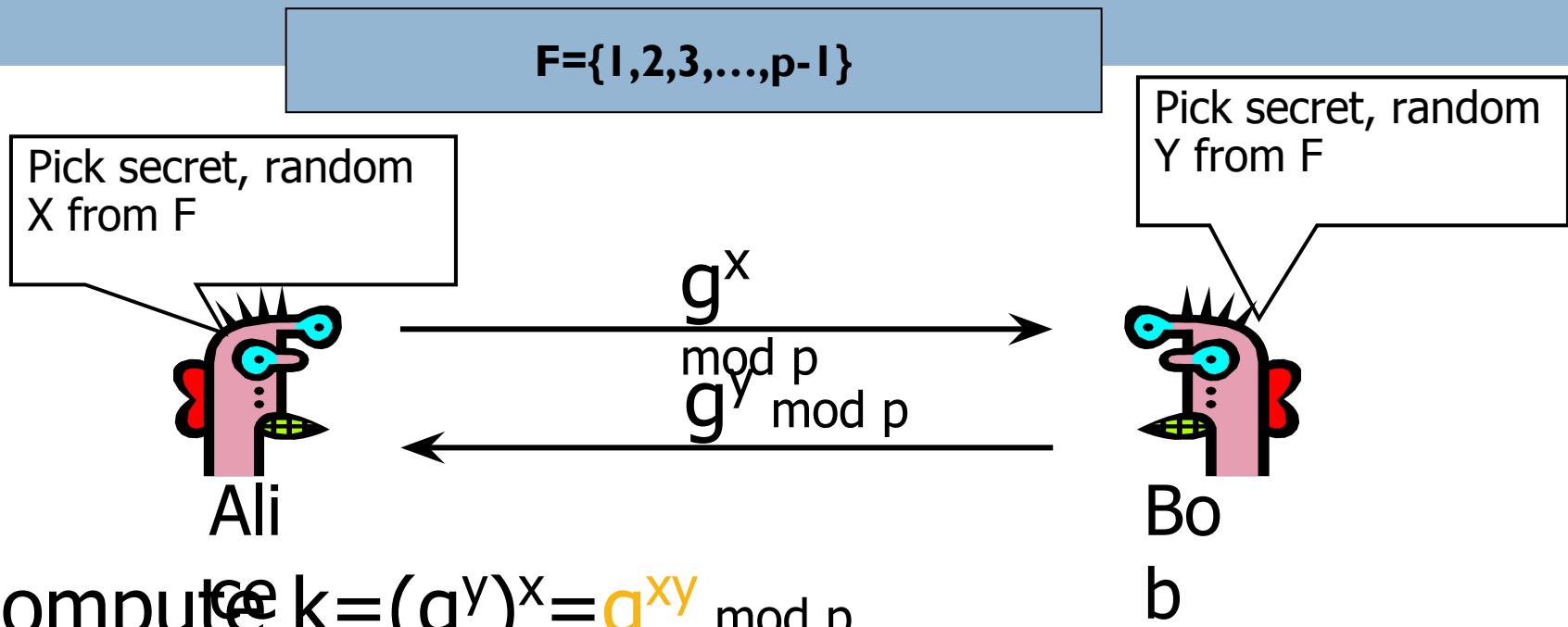
# Elliptic curves over finite fields... : Exercise

- Let  $E$  be the curve  $y^2 + xy = x^3 + 1$  over  $\mathbb{F}_2$ , find all the points on  $E$

*What is the order of  $E(F_2)$ ?*

*Is  $E(F_2)$  cyclic??? If yes,  
what is the generator??*

# Discrete logarithms in Finite Fields



Eve has to compute  $g^{xy}$  from  $g^x$  and  $g^y$  without knowing  $x$  and  $y$ ...  
She faces the **Discrete Logarithm Problem** in finite fields

# Elliptic Curve Discrete Logarithm Problem (ECDLP)

If we are working over a large finite field and are given points  $P$  and  $kP$ , it is computationally hard to determine the value of  $k$ . This is called the **discrete logarithm problem for elliptic curves (ECDLP)** and is the basis for the cryptographic applications.

# What Is Elliptic Curve Cryptography (ECC)?

- Elliptic curve cryptography [ECC] is a **public-key** cryptosystem just like RSA, El Gamal.
- Every user has a **public** and a **private** key.
  - Public key is used for encryption/signature verification.
  - Private key is used for decryption/signature generation.
- Elliptic curves are used as an extension to other current cryptosystems.
  - Elliptic Curve El-Gamal Encryption
  - Elliptic Curve Diffie-Hellman Key Exchange
  - Elliptic Curve Digital Signature Algorithm

# Using Elliptic Curves In Cryptography

- The central part of any cryptosystem involving elliptic curves is the **elliptic group**.
- All public-key cryptosystems have some underlying mathematical operation.
  - RSA has **exponentiation** (raising the message or ciphertext to the public or private values)
  - ECC has **point multiplication** (repeated addition of two points).

# Discrete Logarithm Key pair generation

- A key pair is associated with a set of public domain parameters  $(p, q, g)$ . Here,  $p$  is a prime,  $q$  is a prime divisor of  $p-1$ , and  $g \in [1, p-1]$  has order  $q$

INPUT : DLdomain parameters  $(p, q, g)$ .

OUTPUT : Public key  $y$  and private key  $x$ .

1. Select  $x \in_R [1, q - 1]$ .

2. Compute  $y = g^x \bmod p$

3. Return  $(y, x)$ .

# ECC Key pair generation

- Let  $E$  be an elliptic curve defined over a finite field  $F_p$ .
- Let  $P$  be a point in  $E(F_p)$ , and suppose that  $P$  has prime order  $n$ . Then the cyclic subgroup of  $E(F_p)$  generated by  $P$  is,

$$P = \{O, P, 2P, 3P, \dots, (n-1)P\}.$$

The public domain parameters are : The prime  $p$ , the equation of the elliptic curve  $E$ , and the point  $P$  and its order  $n$  :  
 $(p, E, P, n)$

A private key is an integer  $d$  that is selected uniformly at random from the interval  $[1, n - 1]$ , and the corresponding public key is  $Q = dP$ .

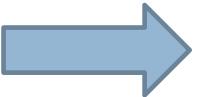
# Basic Elgamal encryption scheme

Basic ElGamal  
Encryption



INPUT : DLdomain parameters  $(p, q, g)$ , public key  $y$ , plaintext  $m \in [0, p - 1]$ .  
OUTPUT : Ciphertext  $(c_1, c_2)$ .  
1. Select  $k \in_R [1, q - 1]$ .  
2. Compute  $c_1 = g^k \bmod p$   
3. Compute  $c_2 = m \cdot y^k \bmod p$   
2.Return  $(c_1, c_2)$ .

Basic ElGamal  
Decryption



INPUT : DLdomain parameters  $(p, q, g)$ , private key  $x$ , ciphertext  $(c_1, c_2)$ .  
OUTPUT : Plaintext  $m$ .  
1. Compute  $m = c_2 \bullet c_1^{-x} \bmod p$ .  
2.Return  $(m)$ .

# ECC Analog to El Gamal : ECEG

EC-ElGamal  
Encryption



INPUT : Elliptic curve domain parameters ( $p, E, P, n$ ), public key  $Q$ , plaintext  $m$ .  
OUTPUT : Ciphertext  $(C_1, C_2)$

1. Represent the message  $m$  as a point  $M$  in  $E(F_p)$
2. Select  $k \in_R [1, n - 1]$ .
3. Compute  $C_1 = kP$ .
4. Compute  $C_2 = M + kQ$ .
5. Return  $(C_1, C_2)$ .

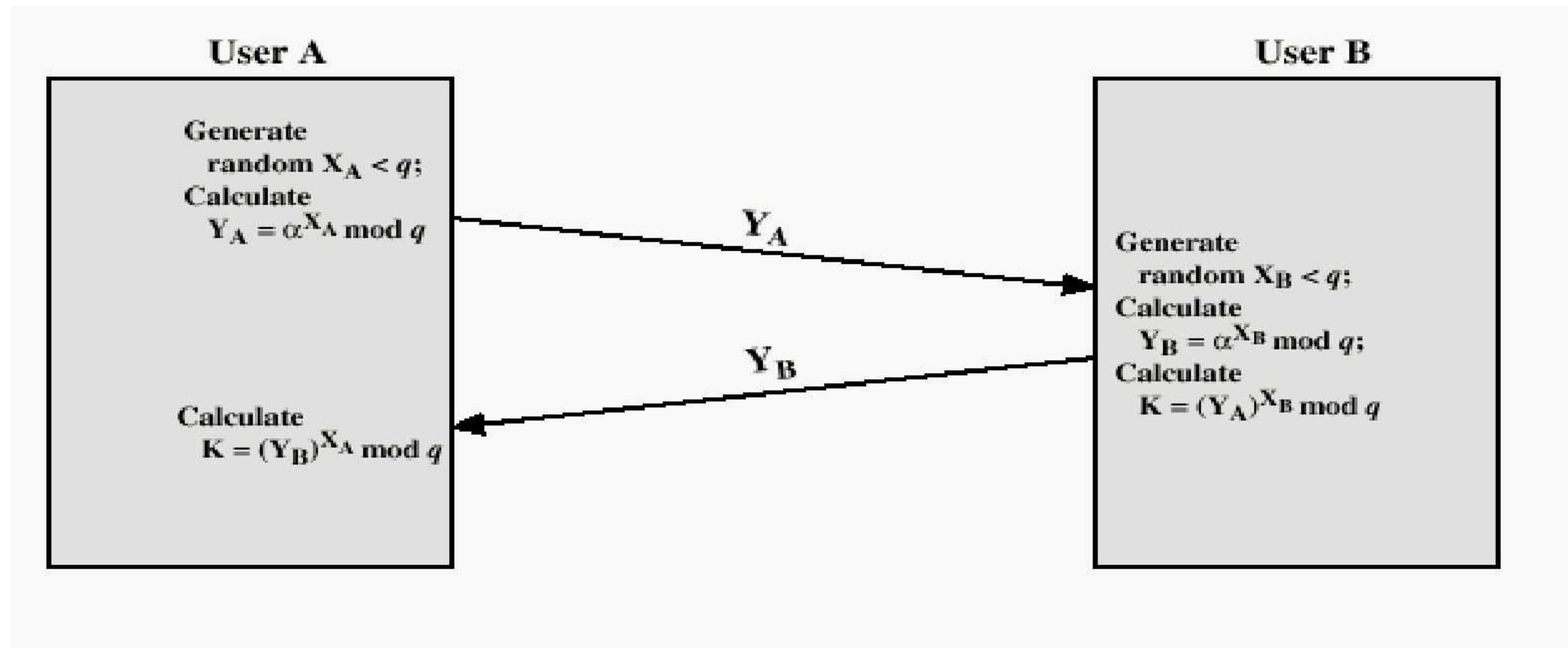
EC-ElGamal  
Decryption



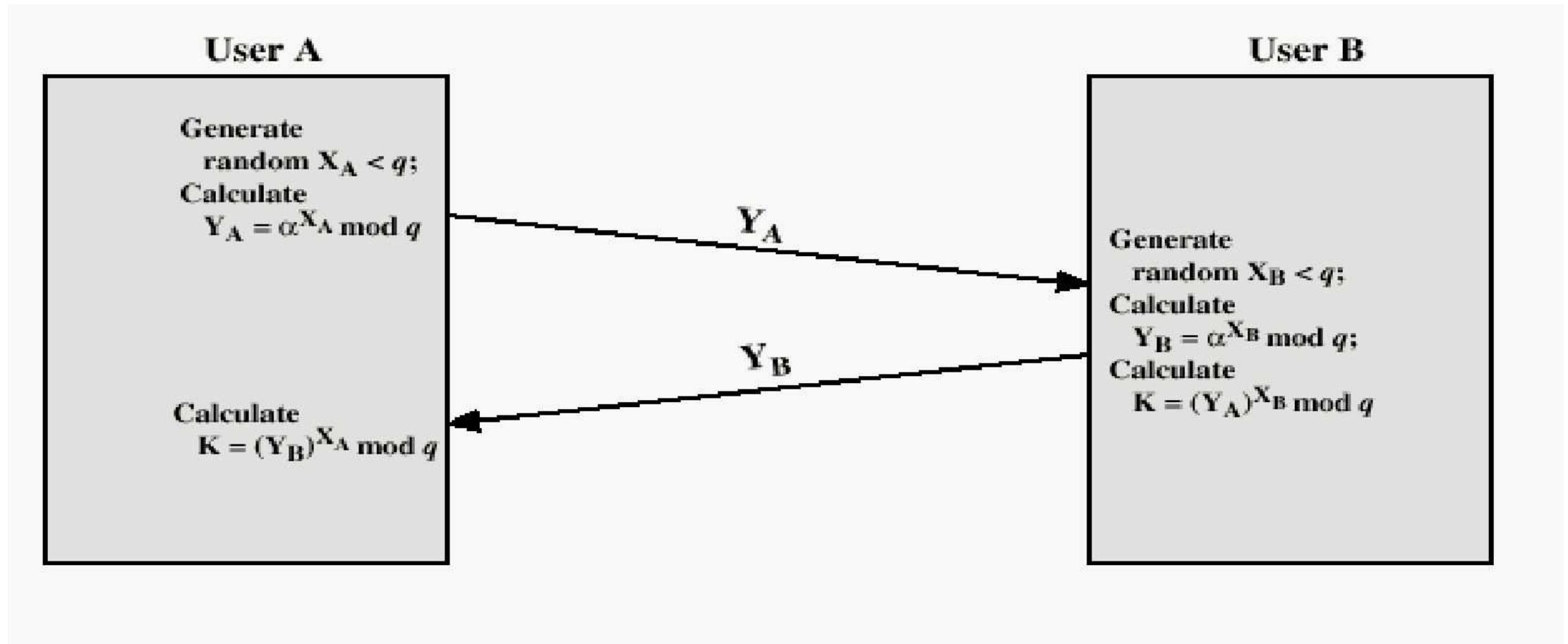
INPUT : Elliptic curve domain parameters ( $p, E, P, n$ ), private key  $d$ , ciphertext  $(C_1, C_2)$   
OUTPUT : Plaintext  $m$ .

1. Compute  $M = C_2 - dC_1$ , and extract  $m$  from  $M$
2. Return  $M$ .

# Diffie-Hellman (DH) Key Exchange



# Can you suggest ECC analog to this????



# ECC Diffie-Hellman: ECDH

- **Public:** Elliptic curve and point  $G=(x,y)$  on curve
- **Secret:** Alice's A and Bob's B



Alice,

A

- Alice computes  $A(B(x,y))$
- Bob computes  $B(A(x,y))$
- These are the same since  $AB = BA$

$A(x,y)$

$B(x,y)$



Bob,

B

## ECC Diffie-Hellman: ECDH...

- **Public:** Curve  $y^2 = x^3 + 7x + b \pmod{37}$  and point  $G = (2, 5)$
- **Alice's secret:**  $A = 4$
- **Bob's secret:**  $B = 7$
- Alice sends Bob:  $4(2, 5) = (7, 32)$
- Bob sends Alice:  $7(2, 5) = (18, 35)$
- Alice computes:  $4(18, 35) = (22, 1)$
- Bob computes:  $7(7, 32) = (22, 1)$

# Digital Signature Algorithm (DSA)

## Signature Generation



**INPUT :** DL domain parameters ( $p, q, g$ ), private key  $x$ , message  $m$ .  
**OUTPUT :** Signature  $(r, s)$ .

1. Select  $k \in_R [1, q - 1]$ .
2. Compute  $T = g^k \text{ mod } p$
3. Compute  $r = T \text{ mod } q$ . If  $r = 0$  then go to step 1.
4. Compute  $h = H(m)$ .
5. Compute  $s = k^{-1}(h + xr) \text{ mod } q$ . If  $s = 0$ , then go to step 1.
6. Return  $(r, s)$ .

**INPUT :** DL domain parameters ( $p, q, g$ ), public key  $y$ , message  $m$ , signature  $(r, s)$ .  
**OUTPUT :** Acceptance or Rejection of a signature.

1. Verify that  $r$  and  $s$  are integers in the interval  $[1, q - 1]$ .  
If any verification fails, then return("Reject the signature").
2. Compute  $h = H(m)$ .
3. Compute  $w = s^{-1} \text{ mod } q$ .
4. Compute  $u_1 = hw \text{ mod } q$  and  $u_2 = rw \text{ mod } q$ .
5. Compute  $T = g^{u_1} y^{u_2} \text{ mod } p$
6. Compute  $r' = T \text{ mod } q$
7. If  $r = r'$  then return ("accept signature");  
else return("reject signature").

## Signature Verification



# ECC analogue to DSA : ECDSA

## Signature Generation



**INPUT :** Domain parameters D, private key d, message m  
**OUTPUT :** Signature (r, s).

1. Select  $k \in_R [1, n - 1]$
2. Compute  $kP = (x_1, y_1)$  and convert  $x_1$  to an integer  $\bar{x}_1$
3. Compute  $r = \bar{x}_1 \bmod n$ . If  $r = 0$  then go to step 1.
4. Compute  $e = H(m)$ .
5. Compute  $s = k^{-1}(e + dr) \bmod n$ . If  $s = 0$  then goto step 1.
6. Return (r, s).

**INPUT :** Domain parameters D, public key Q, message m and Signature (r, s).  
**OUTPUT :** Acceptance or rejection of the signature

1. Verify that r and s are integers in the interval  $[1, n - 1]$ .  
If any verification fails then return ("Reject the signature").
2. Compute  $e = H(m)$ .
3. Compute  $w = s^{-1} \bmod n$
4. Compute  $u_1 = ew \bmod n$  and  $u_2 = rw \bmod n$
5. Compute  $X = u_1P + u_2Q$ .
6. If  $X = \infty$  then return("reject signature").
7. Convert the x - coordinate  $x_1$  to an integer  $\bar{x}_1$ ; compute  $v = \bar{x}_1 \bmod n$ .
8. If  $v = r$  then return("Accept the signature");  
else return("Reject the signature");



## Signature Verification

# Why use ECC?

- Criteria to be considered while selecting PKC for application
  - **Functionality:** Does the public-key family provide the desired capabilities?
  - **Security:** What assurances are available that the protocols are secure?
  - **Performance:** For the desired level of security, do the protocols meet performance objectives?
  - Also some misc. factors such as existence of best-practice standards developed by accredited standards organizations, the availability of commercial cryptographic products, and patent coverage.

# Why use ECC?...

- The RSA, DL and EC families all provide the basic functionality expected of public-key cryptography
- But..... How do we analyze these Cryptosystems?
  - How difficult is the **underlying problem** that it is based upon
    - RSA – Integer Factorization
    - DH – Discrete Logarithms
    - ECC - Elliptic Curve Discrete Logarithm problem

# Why use ECC?...

- How do we measure **difficulty**?
  - We examine the algorithms used to solve these problems
  - Integer factorization
    - Number Field Sieve (NFS) : **Sub exponential** running time
  - Discrete Logarithm
    - Number Field Sieve (NFS) : **Sub exponential** running time
    - Pollard's rho algorithm
  - Elliptic Curve Discrete Logarithm Problem(ECDLP)
    - Pollard's rho algorithm : **Fully exponential** running time

# Why use ECC?...

- To **protect** a 128 bit AES key it would take a:
  - RSA Key Size: 3072 bits
  - ECC Key Size: 256 bits
- How do we strengthen RSA?
  - Increase the key length
- **Impractical?**

NIST guidelines for public key sizes for AES			
ECC KEY SIZE (Bits)	RSA KEY SIZE (Bits)	KEY SIZE RATIO	AES KEY SIZE (Bits)
163	1024	1 : 6	
256	3072	1 : 12	128
384	7680	1 : 20	192
512	15 360	1 : 30	256

# Applications of ECC

- Many devices are small and have limited storage and computational power
- Where can we apply ECC?
  - **Wireless communication devices**
  - Smart cards
  - Web servers that need to handle many encryption sessions
  - **Any application where security is needed but lacks the power, storage and computational power that is necessary for our current cryptosystems**

# Libraries Supporting ECC

Sr. No.	Library	Remarks
1	borZoi	<ul style="list-style-type: none"><li>• It is written in C++</li><li>• Supports ECDSA, ECIES and ECDH.</li></ul>
2	Crypto++	<ul style="list-style-type: none"><li>• C++ library supporting ECDSA, ECDH and ECIES.</li><li>• It supports both binary and prime curves.</li></ul>
3	LibTomCrypt	<ul style="list-style-type: none"><li>• It supports only ECDSA and ECDH not ECIES.</li><li>• It supports only curves defined over prime fields.</li><li>• It has nice interface and documentation.</li></ul>
4	LiDIA	<ul style="list-style-type: none"><li>• It is free for noncommercial use.</li><li>• It supports curves defined over binary and prime fields.</li><li>• In this library points can be represented in either affine or projective coordinates.</li></ul>
5	MIRACL	<ul style="list-style-type: none"><li>• It is also free for nonprofit purposes.</li><li>• It also supports curves defined over prime and binary curves.</li><li>• It is the fastest library and it also supports pre computation.</li></ul>

# Libraries Supporting ECC...

Sr. No.	Library	Remarks
6	OpenSSL	<ul style="list-style-type: none"><li>• open-source written in C.</li><li>• It supports ECDSA and ECDH.</li><li>• It has very poor documentation.</li></ul>
7	Bouncy Castle	<ul style="list-style-type: none"><li>• It supports ECDSA, ECDH and ECIES.</li><li>• It provides supports only for curves defined over prime fields, although the documentation refers to binary curves also.</li><li>• It does not support pre computation.</li></ul>
8	FlexiProvider	<ul style="list-style-type: none"><li>• It supports ECDSA, ECNR, ECIES and also ECDH.</li><li>• It also supports curves defined over binary and prime fields.</li><li>• It does not provide and support for pre computation.</li></ul>
9	IAIK	<ul style="list-style-type: none"><li>• It is available under educational, commercial or open source licenses.</li><li>• It supports ECDSA and ECDH.</li><li>• It can handle curves defined over binary and prime fields.</li><li>• It provides pre computation for prime curves.</li></ul>

# Libraries Supporting ECC...

Sr. No.	Library	Remarks
10	Libecc	<ul style="list-style-type: none"><li>• Libecc is an elliptic curve crypto library for C++ developers.</li><li>• It is currently under development.</li></ul>
11	ECC-LIB	<ul style="list-style-type: none"><li>• fully-equipped, portable, and modular library for Elliptic Curve (EC) Cryptography</li><li>• The library is implemented in ANSI C.</li></ul>
12	PBC Library	<ul style="list-style-type: none"><li>• The PBC (Pairing based cryptography) library is a free C library</li><li>• Performs the mathematical operations underlying pairing-based cryptosystems.</li></ul>



**Thank you !!**

**Q&A**

GLOBAL  
EDITION

# Cryptography and Network Security

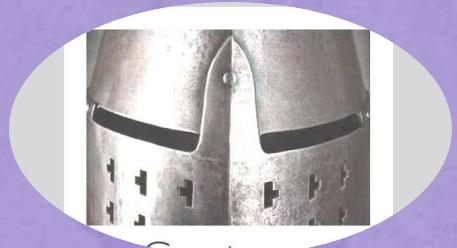
*Principles and Practice*

SEVENTH EDITION

William Stallings



Pearson



Courtesy

# Chapter 14

---

## Key Management and Distribution

# Key Distribution Technique

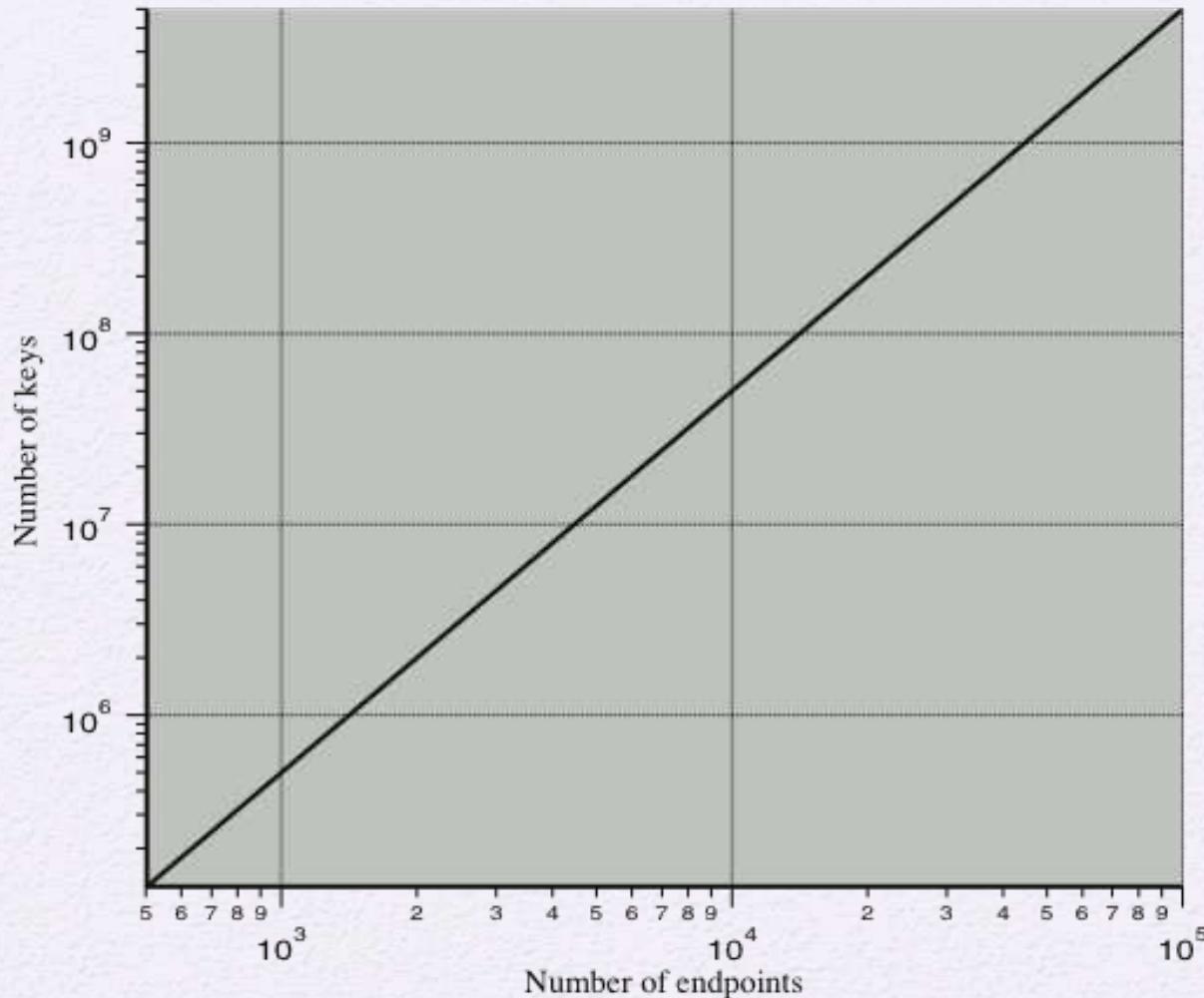
- Term that refers to the means of delivering a key to two parties who wish to exchange data without allowing others to see the key
- For symmetric encryption to work, the two parties to an exchange must share the same key, and that key must be protected from access by others
- Frequent key changes are desirable to limit the amount of data compromised if an attacker learns the key

# Symmetric Key Distribution

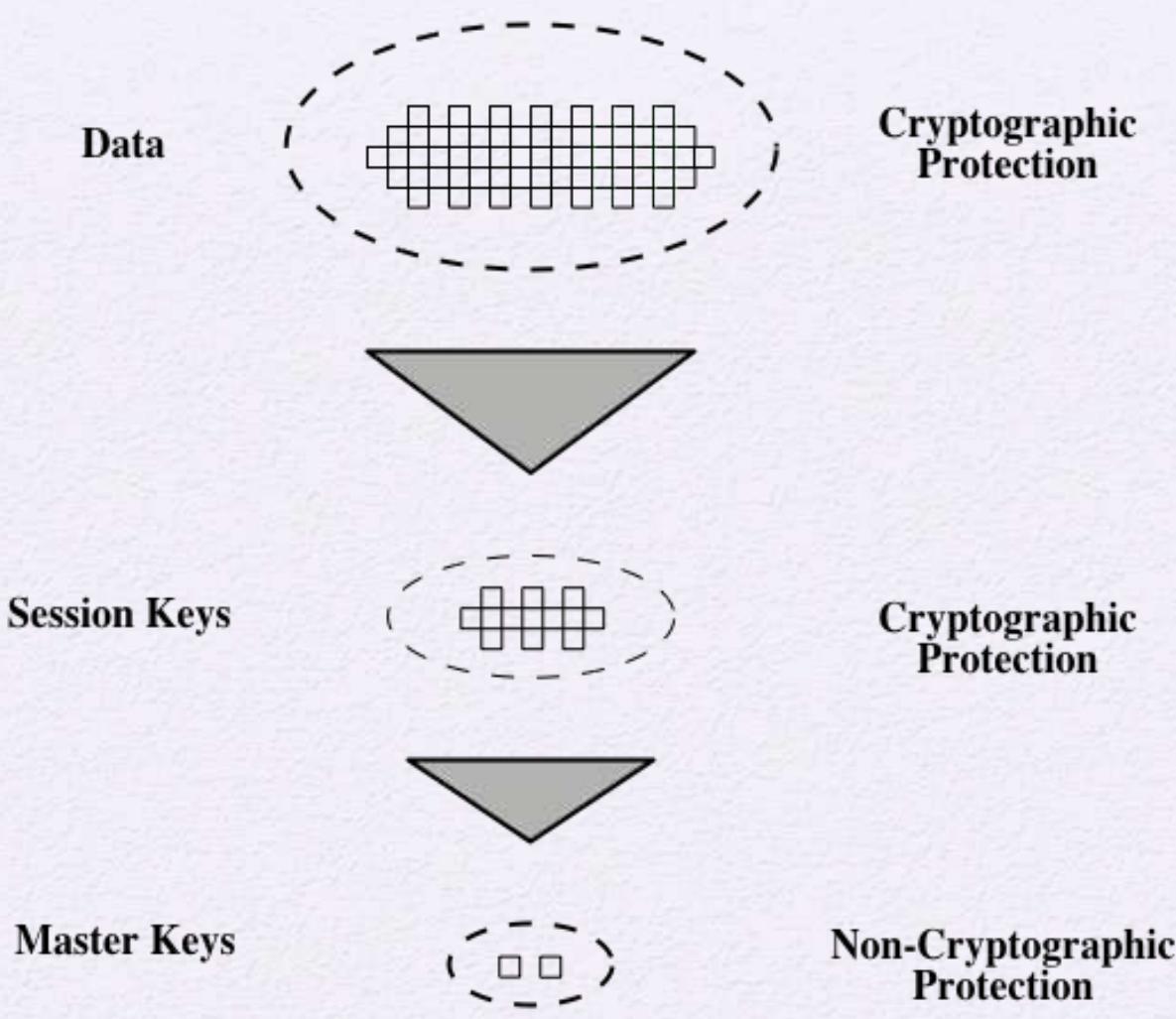
Given parties A and B, key distribution can be achieved in a number of ways:

- A can select a key and physically deliver it to B
- A third party can select the key and physically deliver it to A and B
- If A and B have previously and recently used a key, one party can transmit the new key to the other, encrypted using the old key
- If A and B each has an encrypted connection to a third party C, C can deliver a key on the encrypted links to A and B

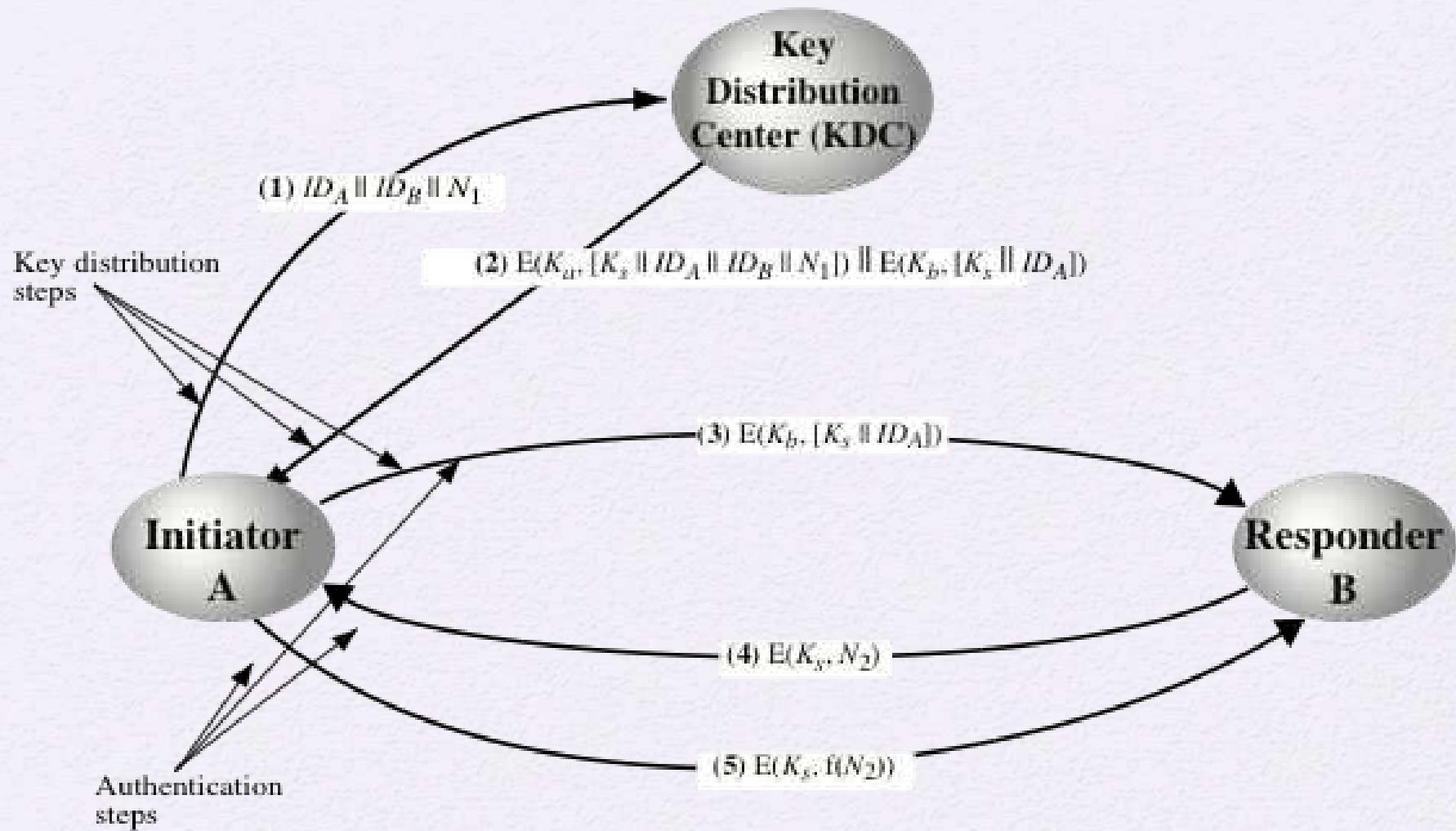




**Figure 14.1 Number of Keys Required to Support Arbitrary Connections Between Endpoints**



**Figure 14.2 The Use of a Key Hierarchy**



**Figure 14.3 Key Distribution Scenario**

# Hierarchical Key Control

- For communication among entities within the same local domain, the local KDC is responsible for key distribution
  - If two entities in different domains desire a shared key, then the corresponding local KDC's can communicate through a global KDC
- The hierarchical concept can be extended to three or more layers
- Scheme minimizes the effort involved in master key distribution because most master keys are those shared by a local KDC with its local entities
  - Limits the range of a faulty or subverted KDC to its local area only

# Session Key Lifetime

For connection-oriented protocols one choice is to use the same session key for the length of time that the connection is open, using a new session key for each new session

A security manager must balance competing considerations:

For a connectionless protocol there is no explicit connection initiation or termination, thus it is not obvious how often one needs to change the session key

The more frequently session keys are exchanged, the more secure they are

The distribution of session keys delays the start of any exchange and places a burden on network capacity

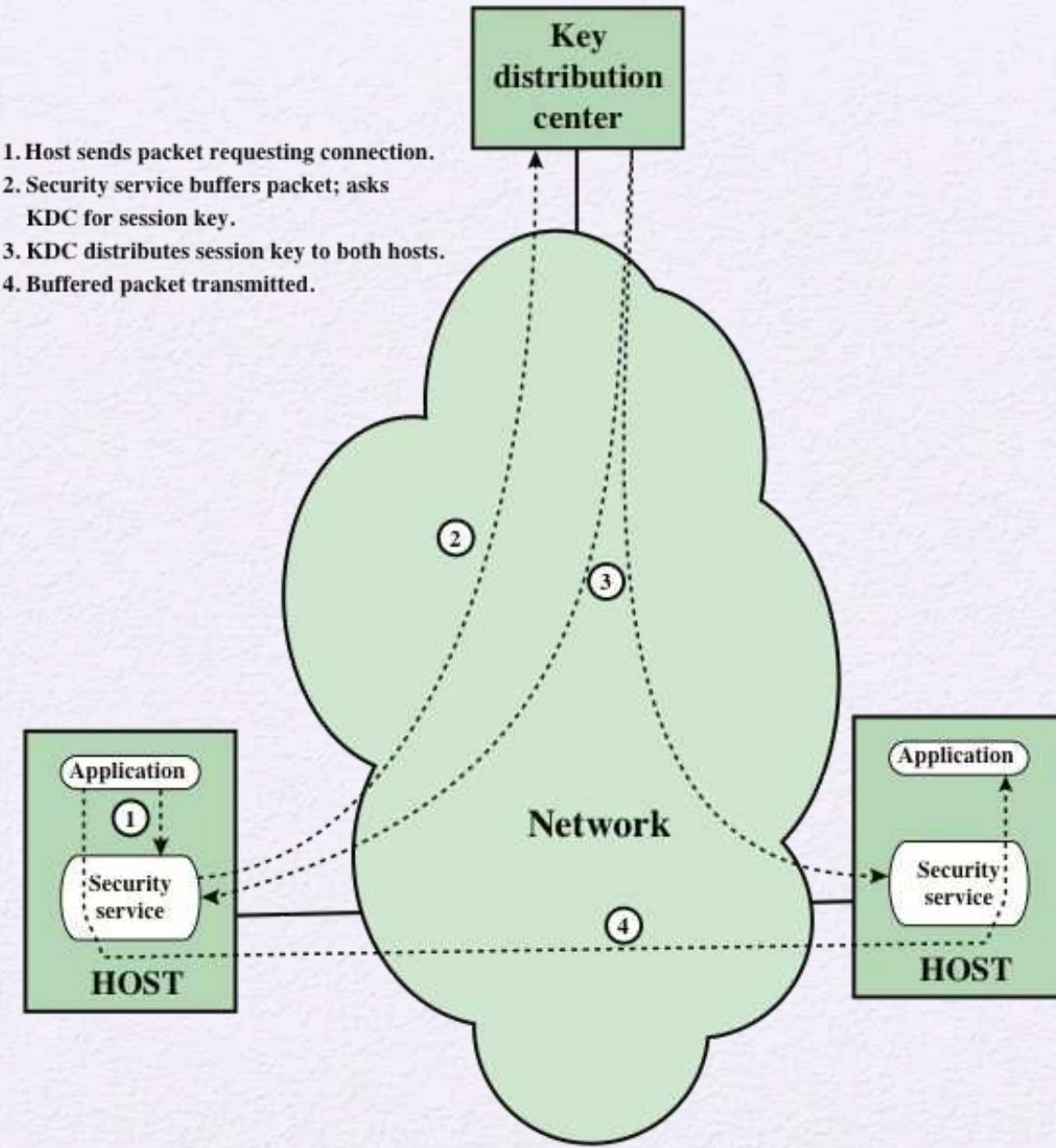
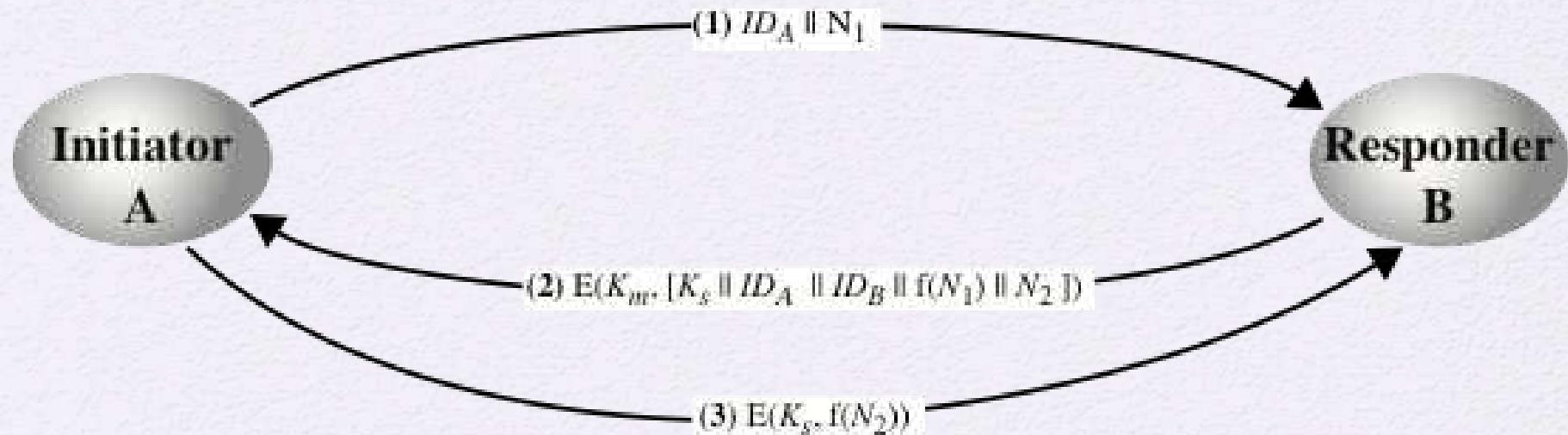


Figure 14.4 Automatic Key Distribution for Connection-Oriented Protocol



**Figure 14.5 Decentralized Key Distribution**

# Controlling Key Usage

- The concept of a key hierarchy and the use of automated key distribution techniques greatly reduce the number of keys that must be manually managed and distributed
- It also may be desirable to impose some control on the way in which automatically distributed keys are used
  - For example, in addition to separating master keys from session keys, we wish to define different types session keys on the basis of use



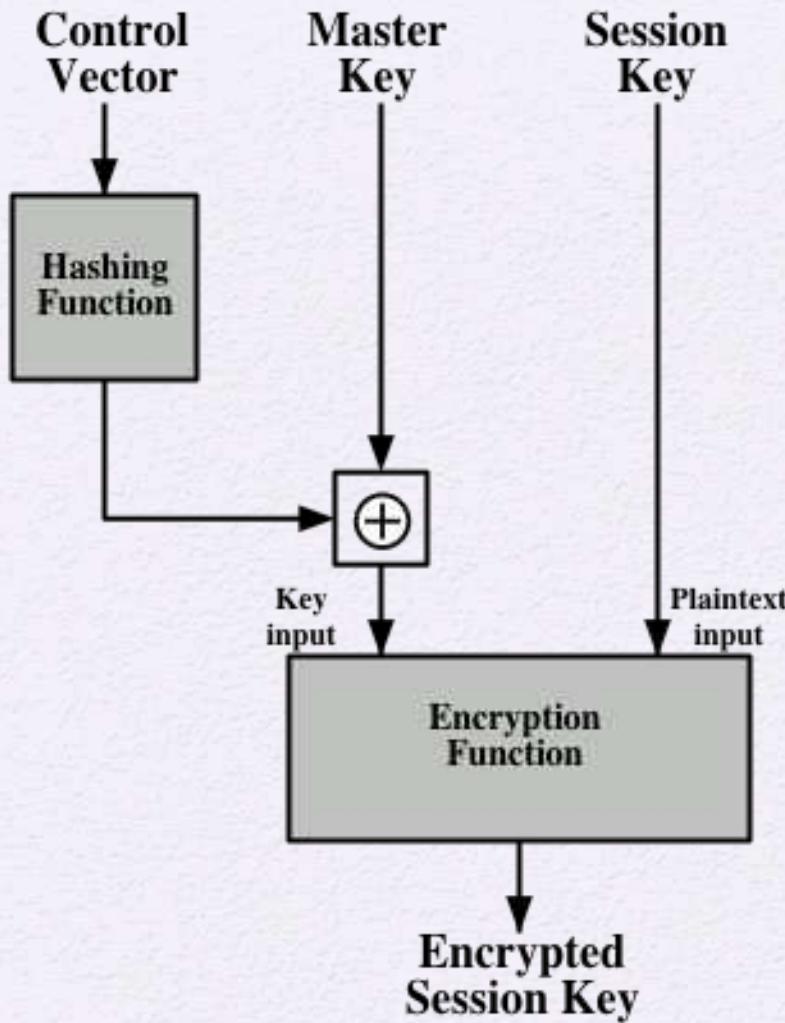
# Key Controls

- Associate a tag with each key
  - For use with DES and makes use of the extra 8 bits in each 64-bit DES key
  - The eight non-key bits ordinarily reserved for parity checking form the key tag
  - Because the tag is embedded in the key, it is encrypted along with the key when that key is distributed, thus providing protection

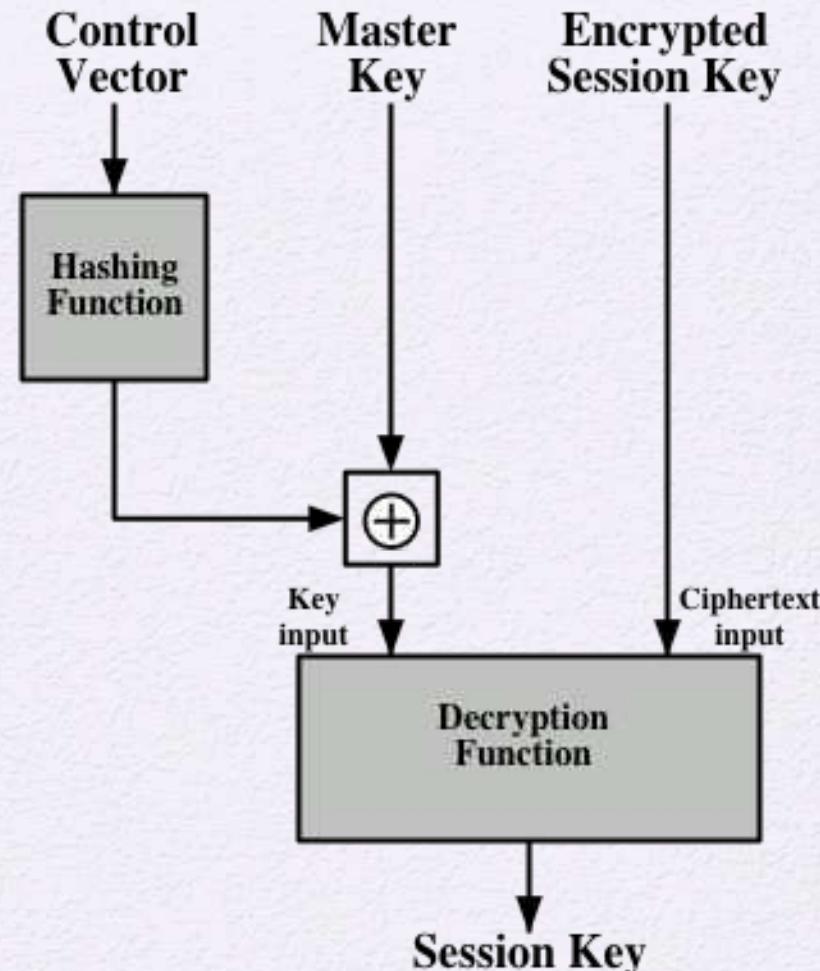
- Drawbacks
  - The tag is limited to 8 bits, limiting its flexibility and functionality
  - Because the tag is not transmitted in clear form, it can be used only at the point of distribution, limiting the ways in which key use can be controlled



• The tag is limited to 8 bits, limiting its flexibility and functionality  
• Because the tag is not transmitted in clear form, it can be used only at the point of distribution, limiting the ways in which key use can be controlled

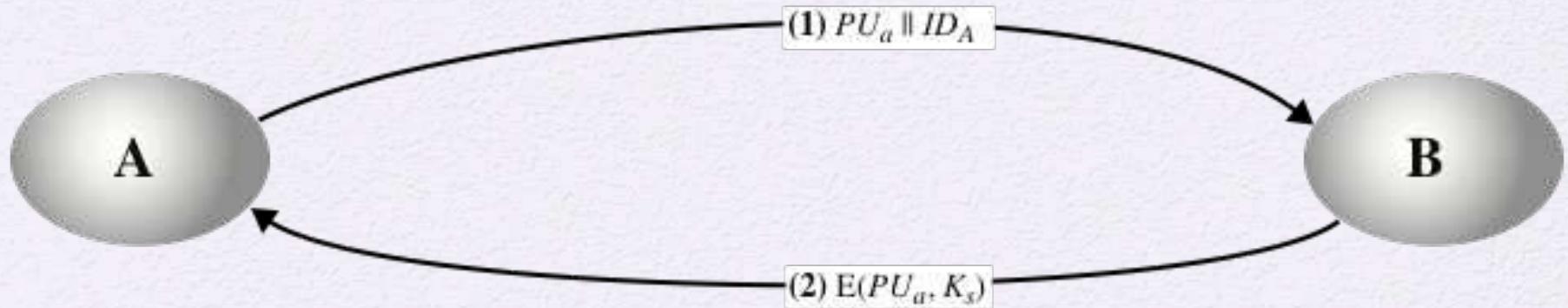


(a) Control Vector Encryption



(b) Control Vector Decryption

**Figure 14.6 Control Vector Encryption and Decryption**



**Figure 14.7 Simple Use of Public-Key Encryption to Establish a Session Key**

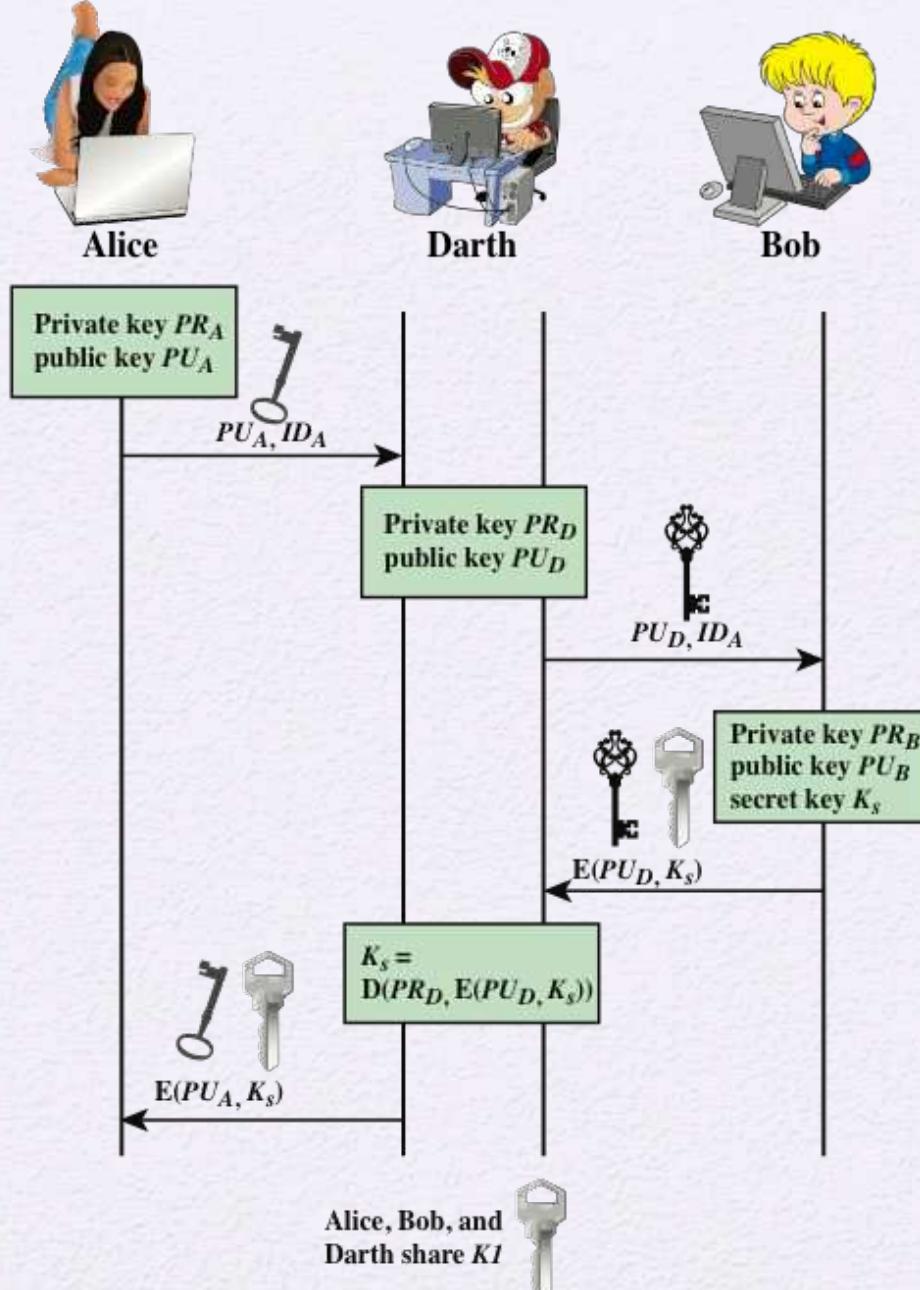
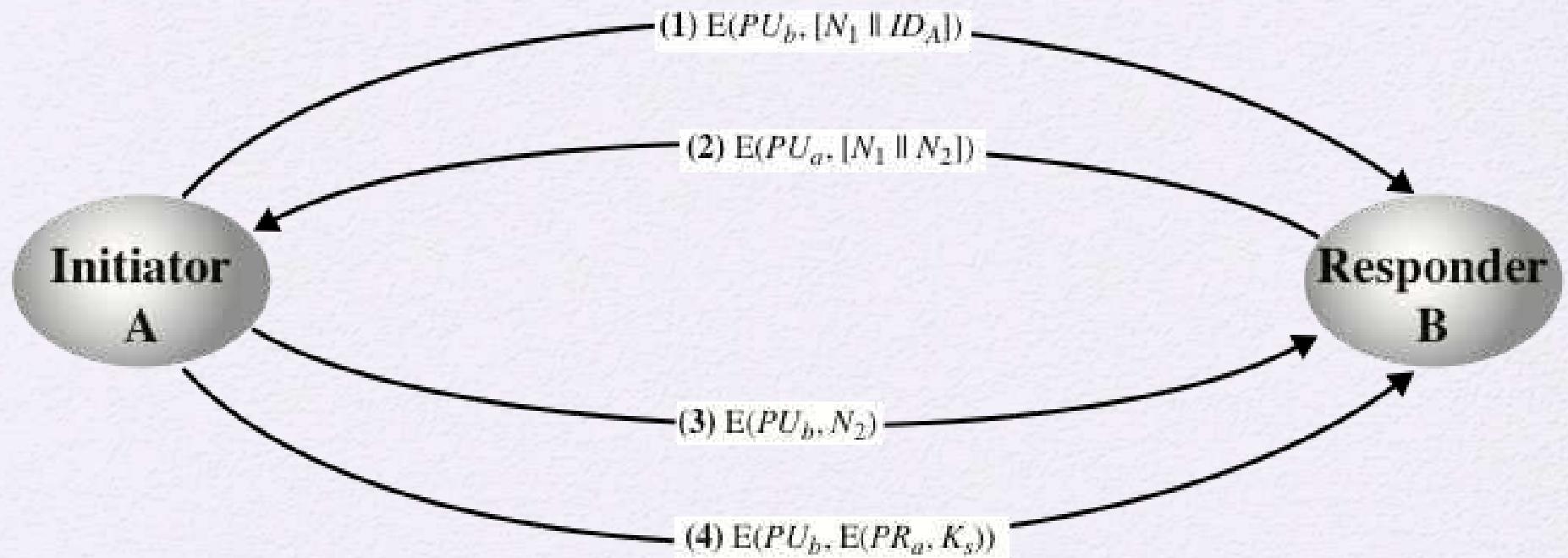


Figure 14.8 Another Man-in-the-Middle Attack



**Figure 14.9** Public-Key Distribution of Secret Keys

# A Hybrid Scheme

- In use on IBM mainframes
- Retains the use of a key distribution center (KDC) that shares a secret master key with each user and distributes secret session keys encrypted with the master key
- A public-key scheme is used to distribute the master keys

Rationale:

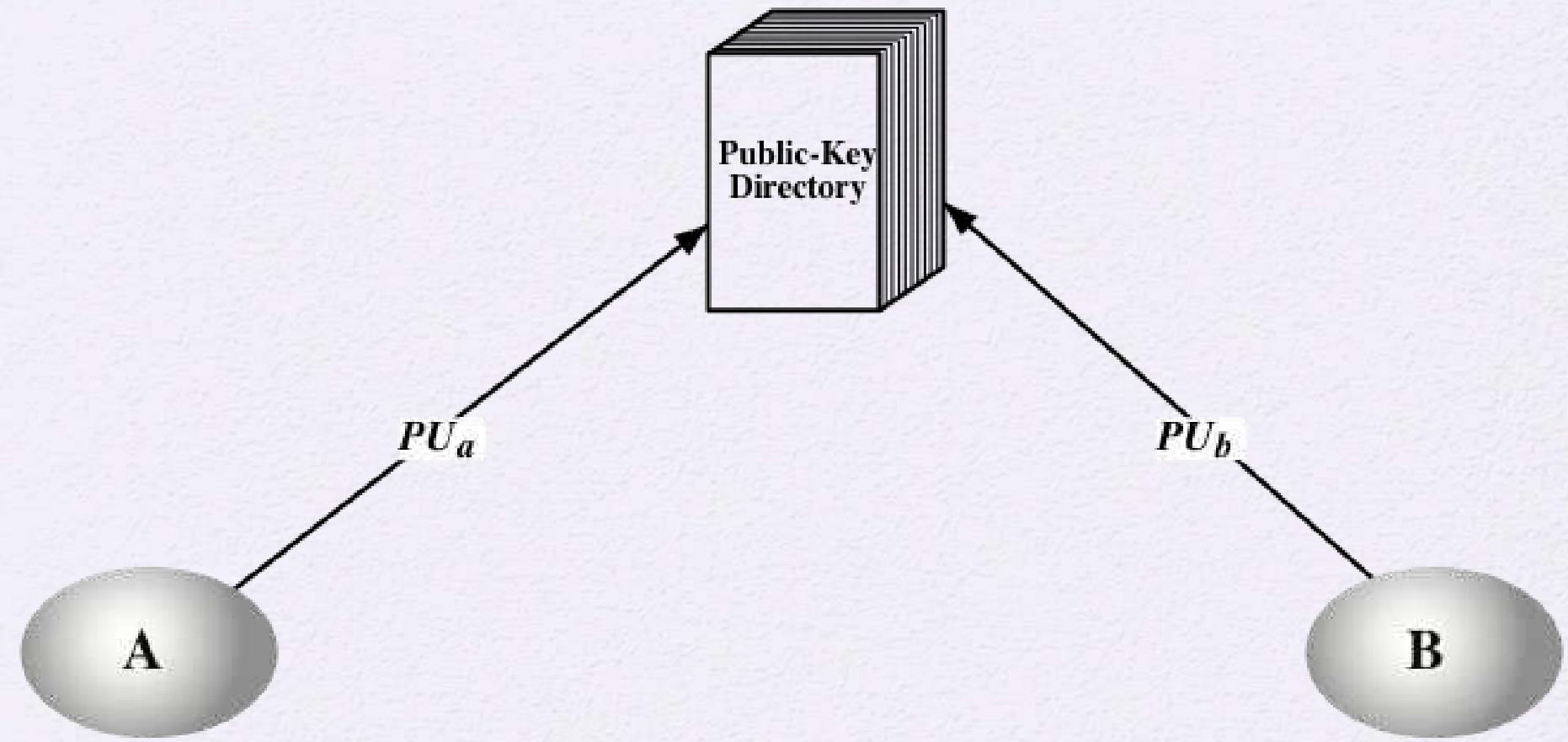
- Performance
- Backward compatibility

# Distribution of Public Keys

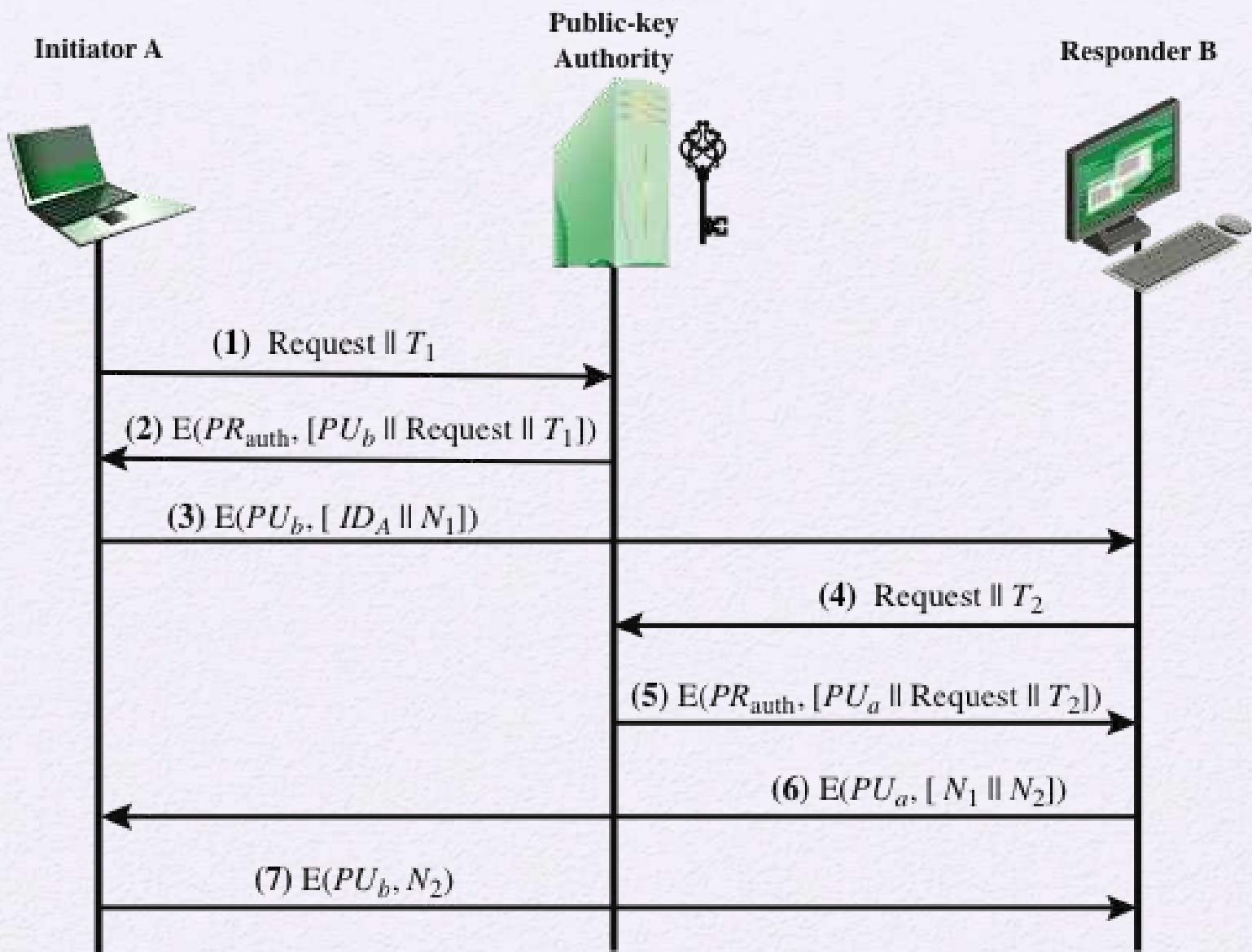
- Several techniques have been proposed for the distribution of public keys.  
Virtually all these proposals can be grouped into the following general schemes:
  - **Public announcement**
  - **Publicly available directory**
  - **Public-key authority**
  - **Public-key certificates**



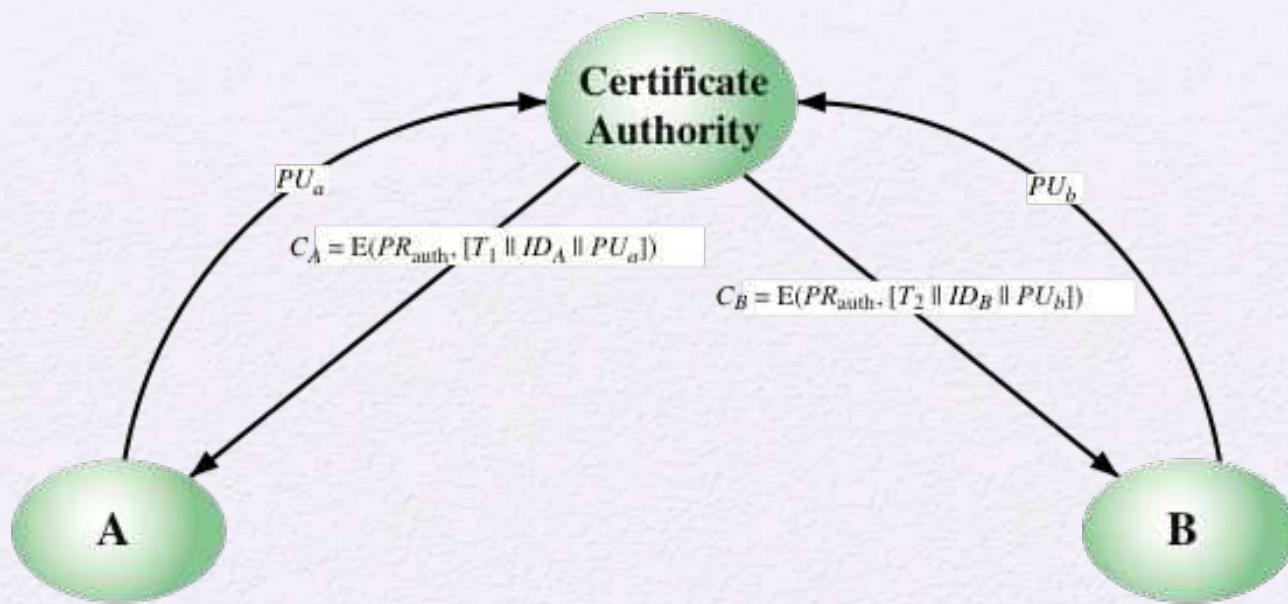
**Figure 14.10 Uncontrolled Public Key Distribution**



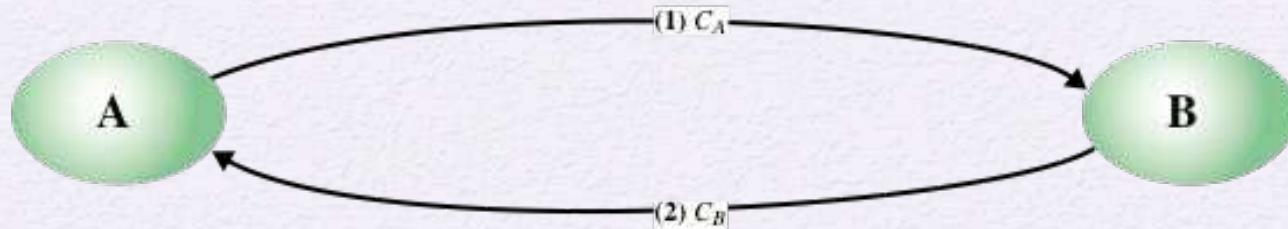
**Figure 14.11 Public Key Publication**



**Figure 14.12 Public-Key Distribution Scenario**



(a) Obtaining certificates from CA

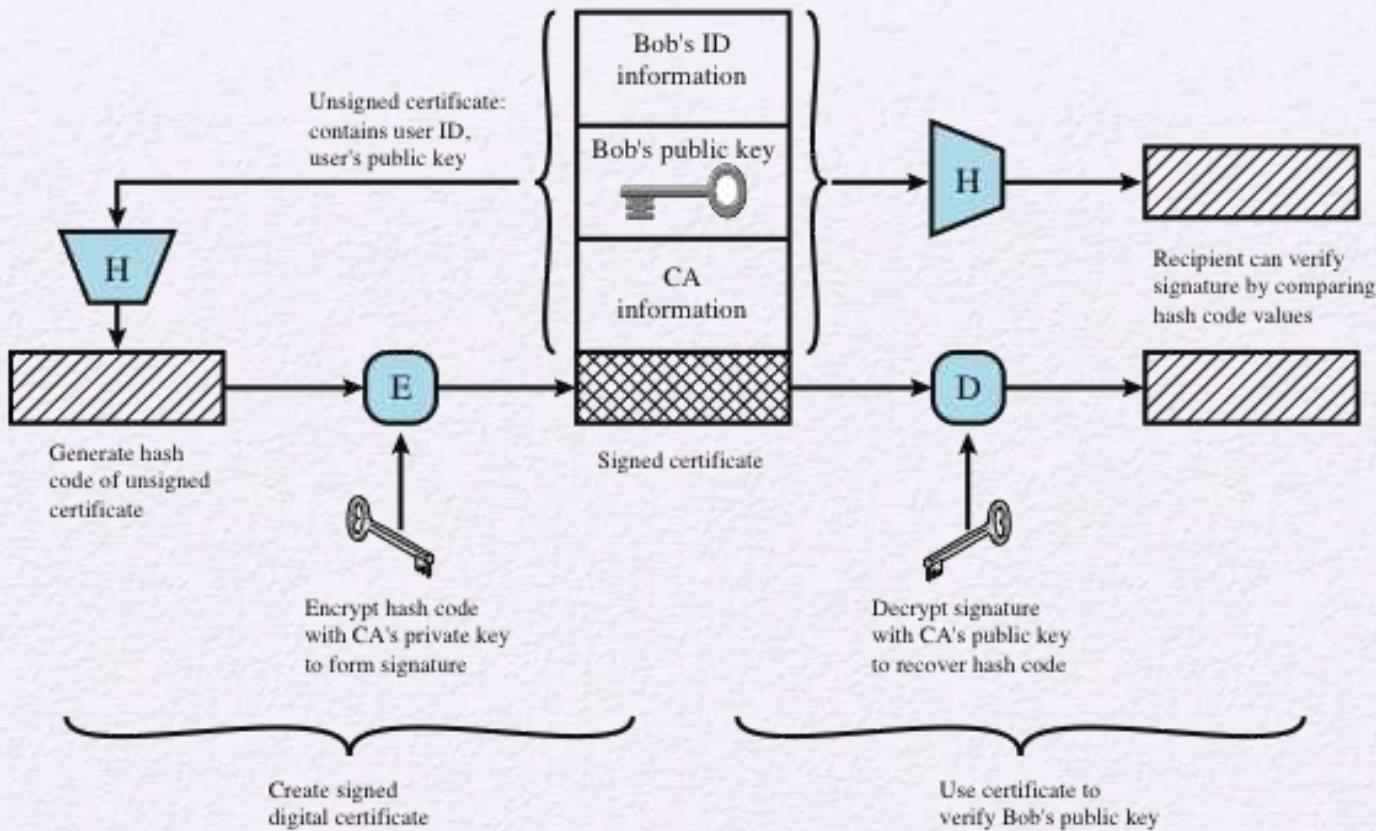


(b) Exchanging certificates

**Figure 14.13 Exchange of Public-Key Certificates**

# X.509 Certificates

- Part of the X.500 series of recommendations that define a directory service
  - The directory is, in effect, a server or distributed set of servers that maintains a database of information about users
- X.509 defines a framework for the provision of authentication services by the X.500 directory to its users
  - Was initially issued in 1988 with the latest revision in 2012
  - Based on the use of public-key cryptography and digital signatures
  - Does not dictate the use of a specific algorithm but recommends RSA
  - Does not dictate a specific hash algorithm
- Each certificate contains the public key of a user and is signed with the private key of a trusted certification authority
- X.509 defines alternative authentication protocols based on the use of public-key certificates



**Figure 14.14 Public-Key Certificate Use**

# Certificates

---

Created by a  
trusted  
Certification  
Authority (CA)  
and have the  
following  
elements:

- Version
- Serial number
- Signature algorithm identifier
- Issuer name
- Period of validity
- Subject name
- Subject's public-key information
- Issuer unique identifier
- Subject unique identifier
- Extensions
- Signature

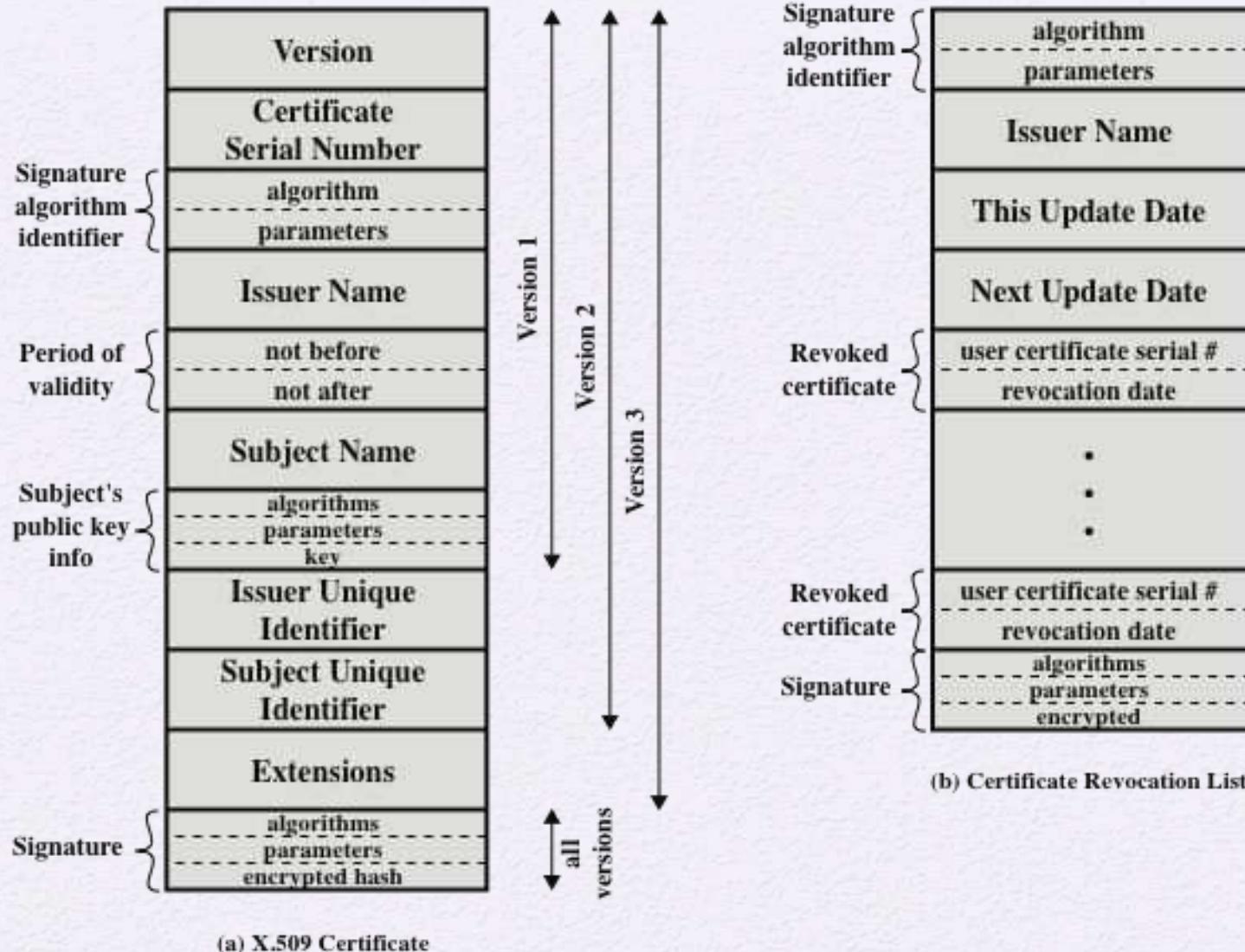


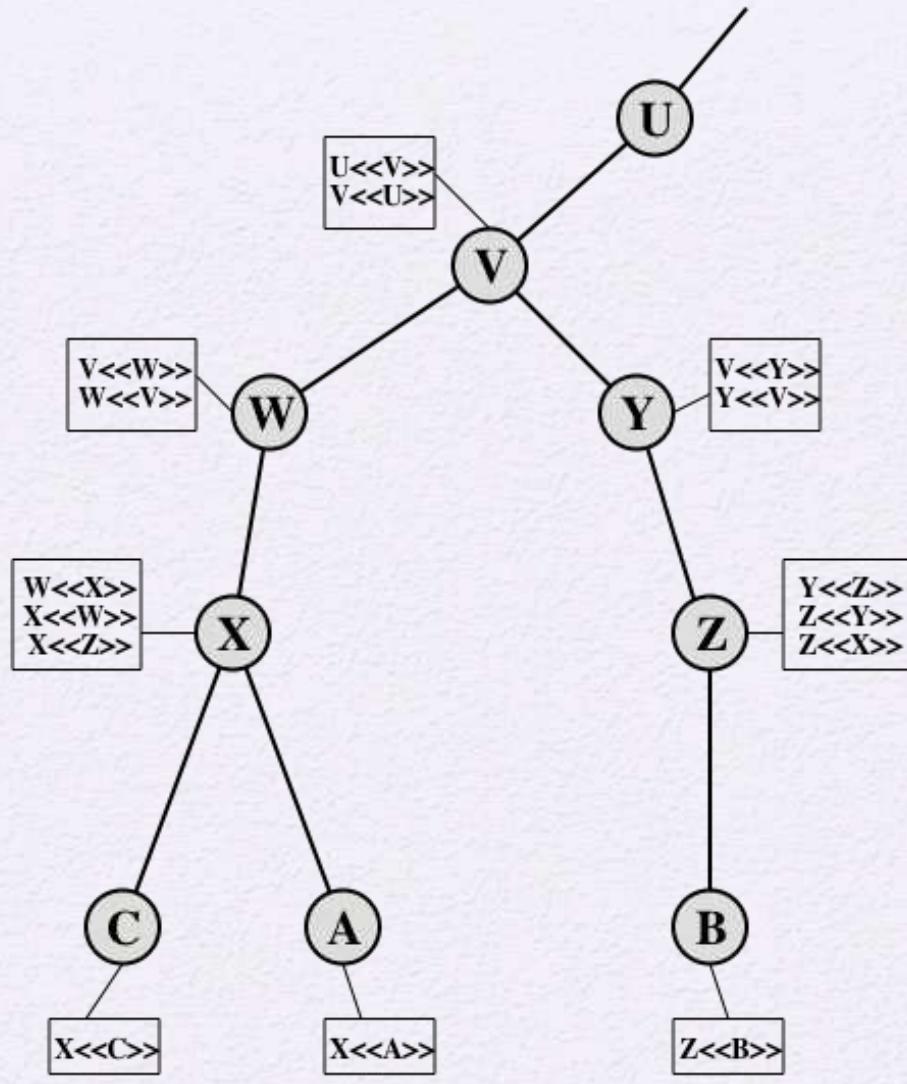
Figure 14.15 X.509 Formats

# Obtaining a Certificate

User certificates generated by a CA have the following characteristics:

- Any user with access to the public key of the CA can verify the user public key that was certified
- No party other than the certification authority can modify the certificate without this being detected

- Because certificates are unforgeable, they can be placed in a directory without the need for the directory to make special efforts to protect them
  - In addition, a user can transmit his or her certificate directly to other users
- Once B is in possession of A's certificate, B has confidence that messages it encrypts with A's public key will be secure from eavesdropping and that messages signed with A's private key are unforgeable



**Figure 14.16 X.509 CA Hierarchy: a Hypothetical Example**

# Certificate Revocation

- Each certificate includes a period of validity
  - Typically a new certificate is issued just before the expiration of the old one
- It may be desirable on occasion to revoke a certificate before it expires, for one of the following reasons:
  - The user's private key is assumed to be compromised
  - The user is no longer certified by this CA
  - The CA's certificate is assumed to be compromised
- Each CA must maintain a list consisting of all revoked but not expired certificates issued by that CA
  - These lists should be posted on the directory

# X.509 Version 3

- Version 2 format does not convey all of the information that recent design and implementation experience has shown to be needed
- Rather than continue to add fields to a fixed format, standards developers felt that a more flexible approach was needed
  - Version 3 includes a number of optional extensions
- The certificate extensions fall into three main categories:
  - Key and policy information
  - Subject and issuer attributes
  - Certification path constraints

Each extension consists of:

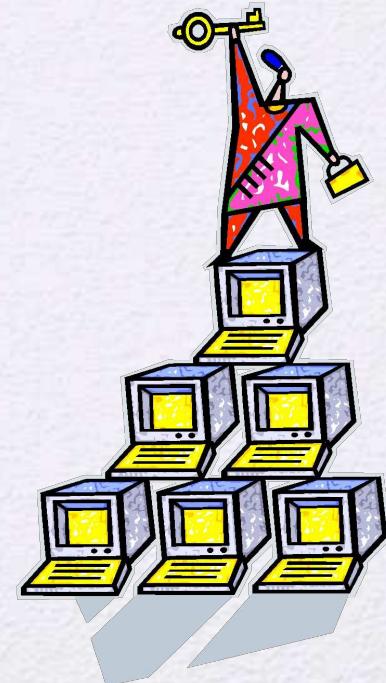
- **An extension identifier**
- **A criticality indicator**
- **An extension value**

# Key and Policy Information

- These extensions convey additional information about the subject and issuer keys plus indicators of certificate policy
- A certificate policy is a named set of rules that indicates the applicability of a certificate to a particular community and/or class of application with common security requirements

Included are:

- Authority key identifier
- Subject key identifier
- Key usage
- Private-key usage period
- Certificate policies
- Policy mappings



# Certificate Subject and Issuer Attributes

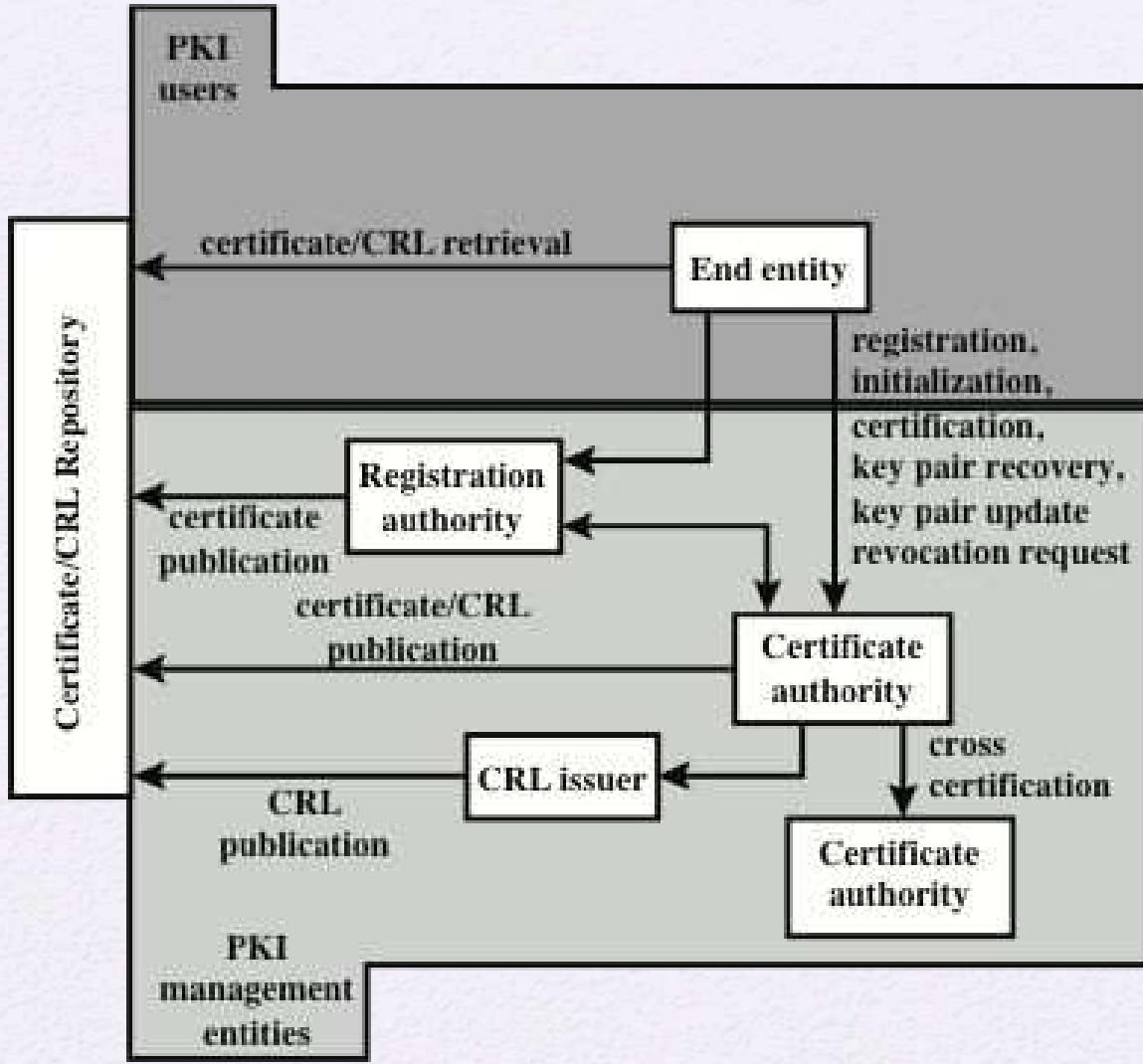
- These extensions support alternative names, in alternative formats, for a certificate subject or certificate issuer
- Can convey additional information about the certificate subject to increase a certificate user's confidence that the certificate subject is a particular person or entity
- The extension fields in this area include:
  - Subject alternative name
  - Issuer alternative name
  - Subject directory attributes



# Certification Path Constraints

- These extensions allow constraint specifications to be included in certificates issued for CAs by other CAs
- The constraints may restrict the types of certificates that can be issued by the subject CA or that may occur subsequently in a certification chain
- The extension fields in this area include:
  - Basic constraints
  - Name constraints
  - Policy constraints





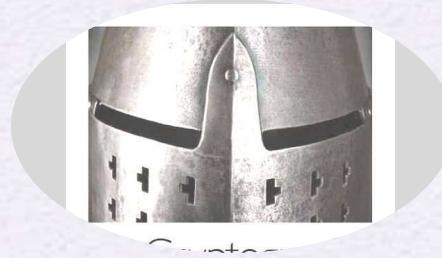
**Figure 14.17 PKIX Architectural Model**

# PKIX Management Functions

- PKIX identifies a number of management functions that potentially need to be supported by management protocols:
  - Registration
  - Initialization
  - Certification
  - Key pair recovery
  - Key pair update
  - Revocation request
  - Cross certification

# Summary

- Symmetric key distribution using symmetric encryption
  - Key distribution scenario
  - Hierarchical key control
  - Session key lifetime
  - Transparent key control scheme
  - Decentralized key control
  - Controlling key usage
- Symmetric key distribution using asymmetric encryption
  - Simple secret key distribution
  - Secret key distribution with confidentiality and authentication
  - Hybrid scheme



- Distribution of public keys
  - Public announcement of public keys
  - Publicly available directory
  - Public-key authority
  - Public-key certificates
- X.509 Certificates
  - X.509 Version 3
- Public-key infrastructure
  - PKIX management functions
  - PKIX management protocols



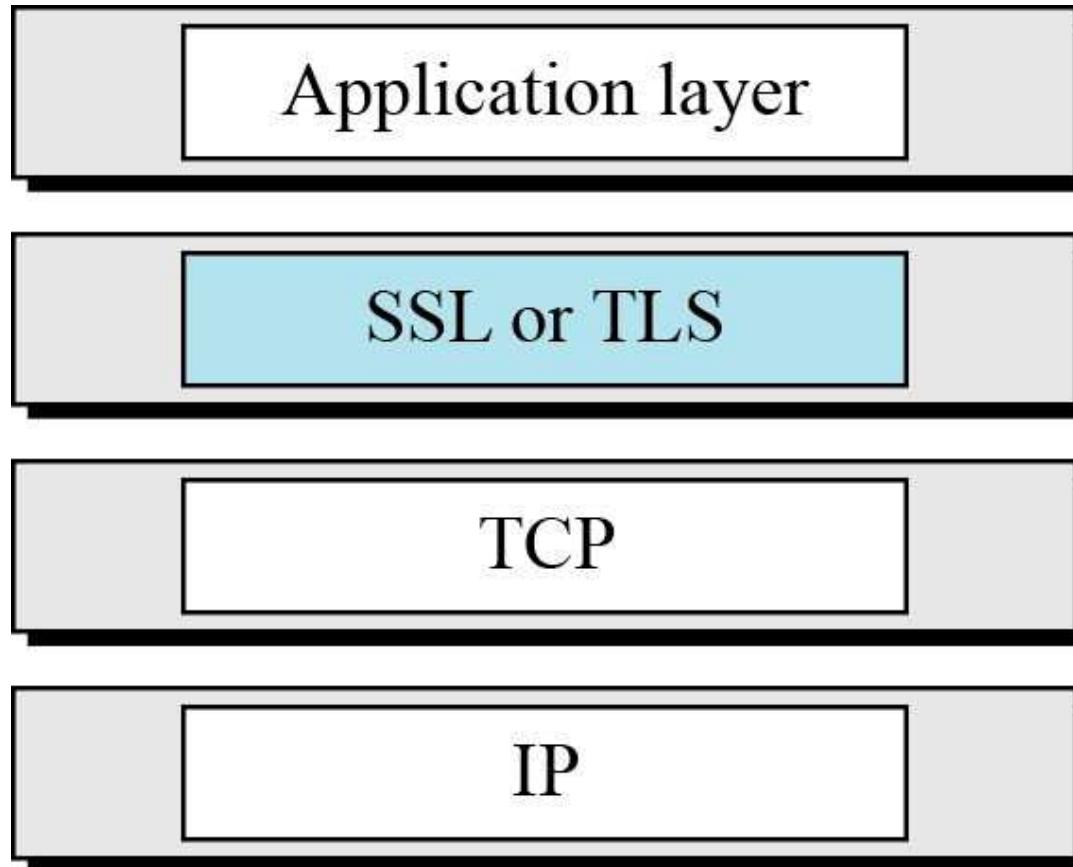
## Chapter 17

### *Security at the Transport Layer: SSL and TLS*

- To discuss the need for security services at the transport layer of the Internet model
- To discuss the general architecture of SSL
- To discuss the general architecture of TLS
- To compare and contrast SSL and TLS

# 17 Continued

**Figure 17.1 Location of SSL and TLS in the Internet model**



# 17-1 SSL ARCHITECTURE

*SSL is designed to provide security and compression services to data generated from the application layer.*

## Topics discussed in this section:

17.1.1 Services

17.1.2 Key Exchange Algorithms

17.1.3 Encryption/Decryption Algorithms

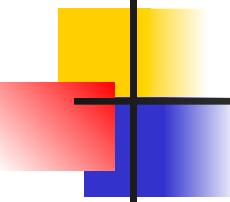
17.1.4 Hash Algorithms

17.1.5 Cipher Suite

17.1.6 Compression Algorithms

17.1.7 Cryptography Parameter Generation

17.1.8 Session and Connections



## *17.1.1 Services*

**Fragmentation**

**Compression**

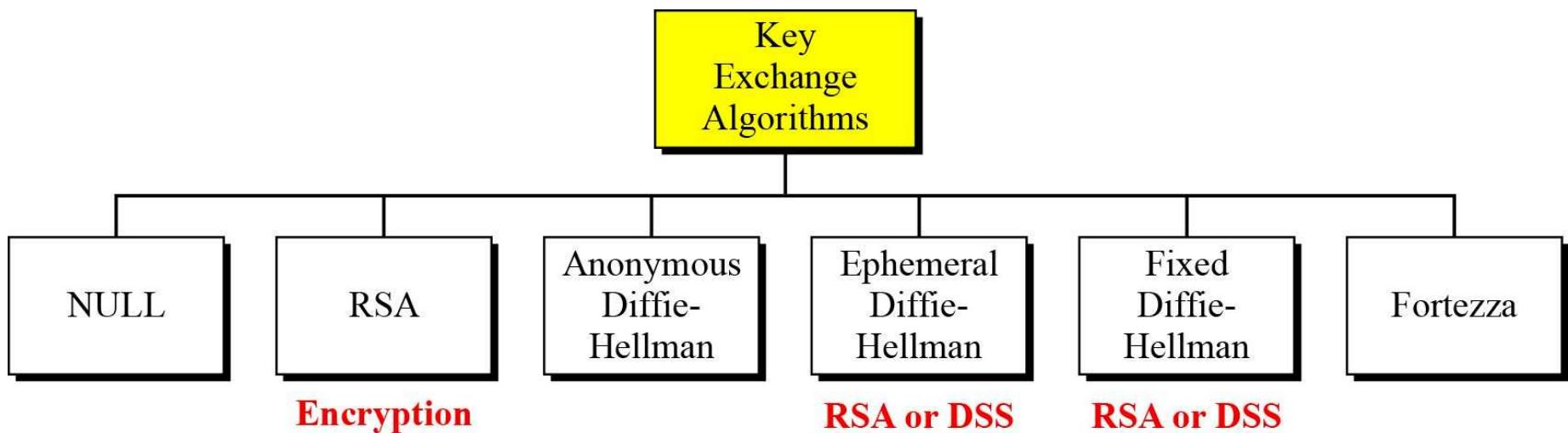
**Message Integrity**

**Confidentiality**

**Framing**

## 17.1.2 Key Exchange Algorithms

Figure 17.2 Key-exchange methods



## **17.1.2 Continued**

**Null**

*There is no key exchange in this method. No pre-master secret is established between the client and the server.*

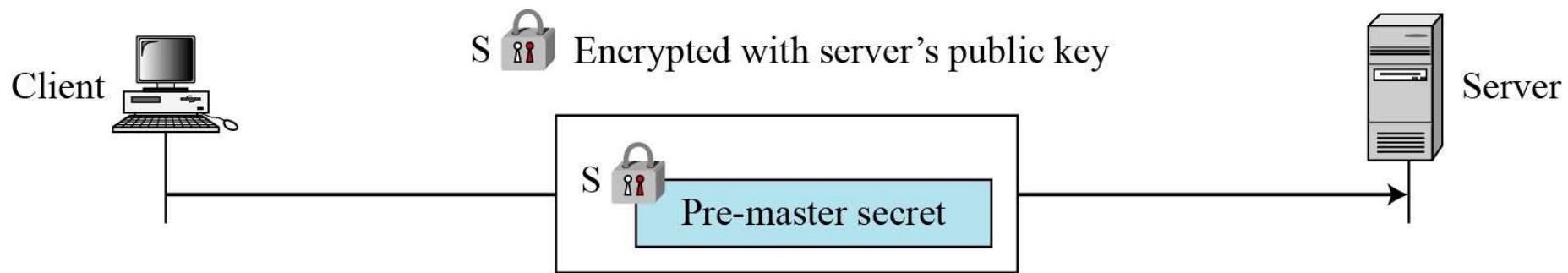
**Note**

**Both client and server need to know the value of the pre-master secret.**

## 17.1.2 *Continued*

### RSA

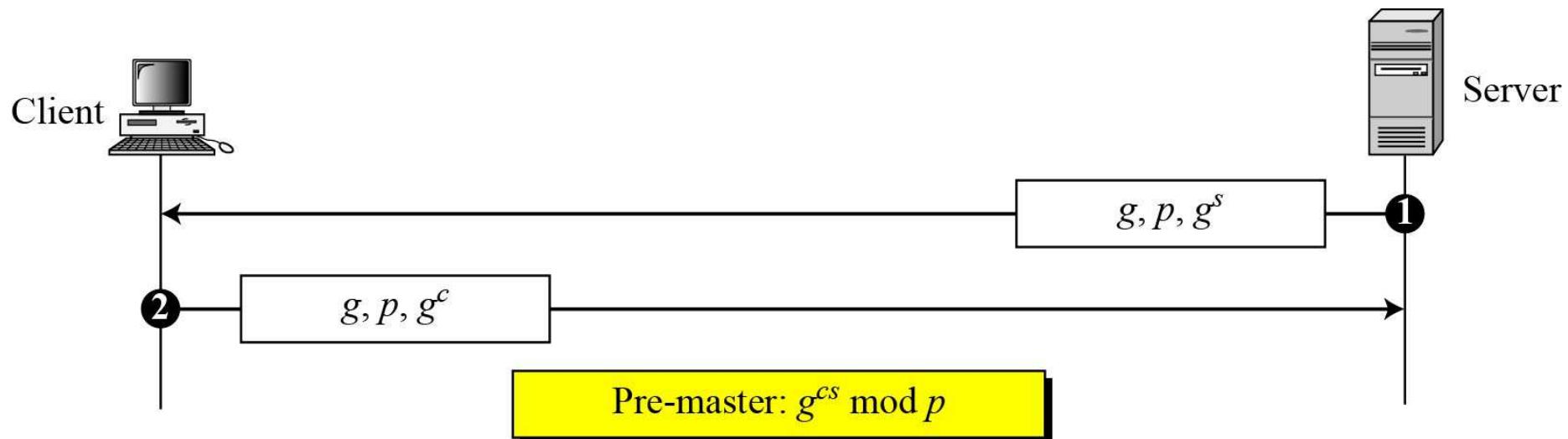
**Figure 17.3 RSA key exchange; server public key**



## 17.1.2 Continued

### Anonymous Diffie-Hellman

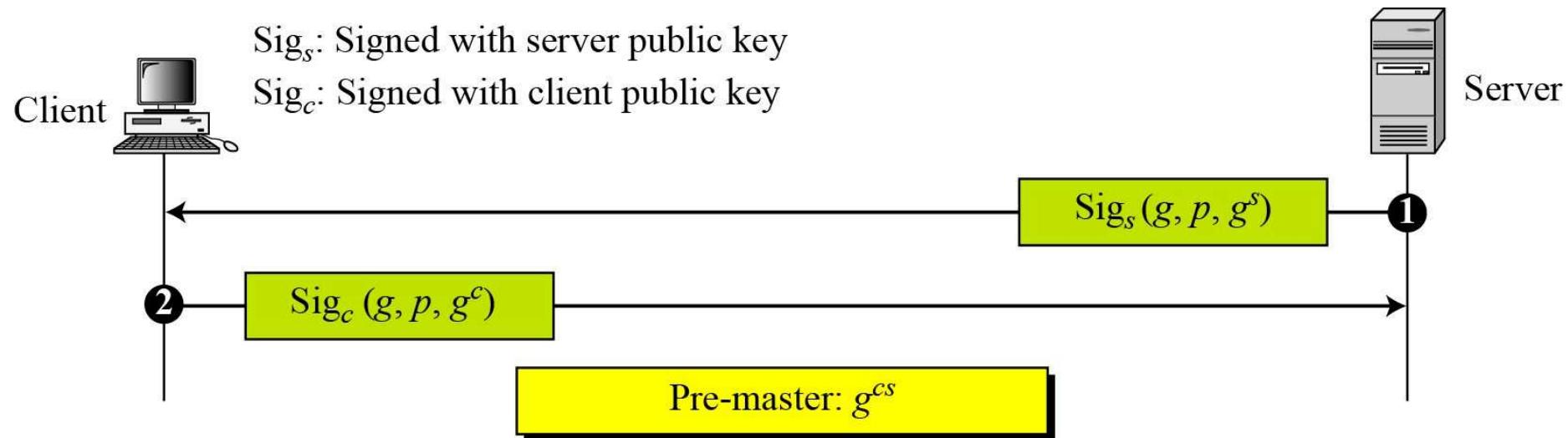
Figure 17.4 *Anonymous Diffie-Hellman key exchange*



## 17.1.2 *Continued*

### Ephemeral Diffie-Hellman key exchange

**Figure 17.5** *Ephemeral Diffie-Hellman key exchange*



## **17.1.2 Continued**

### **Fixed Diffie-Hellman**

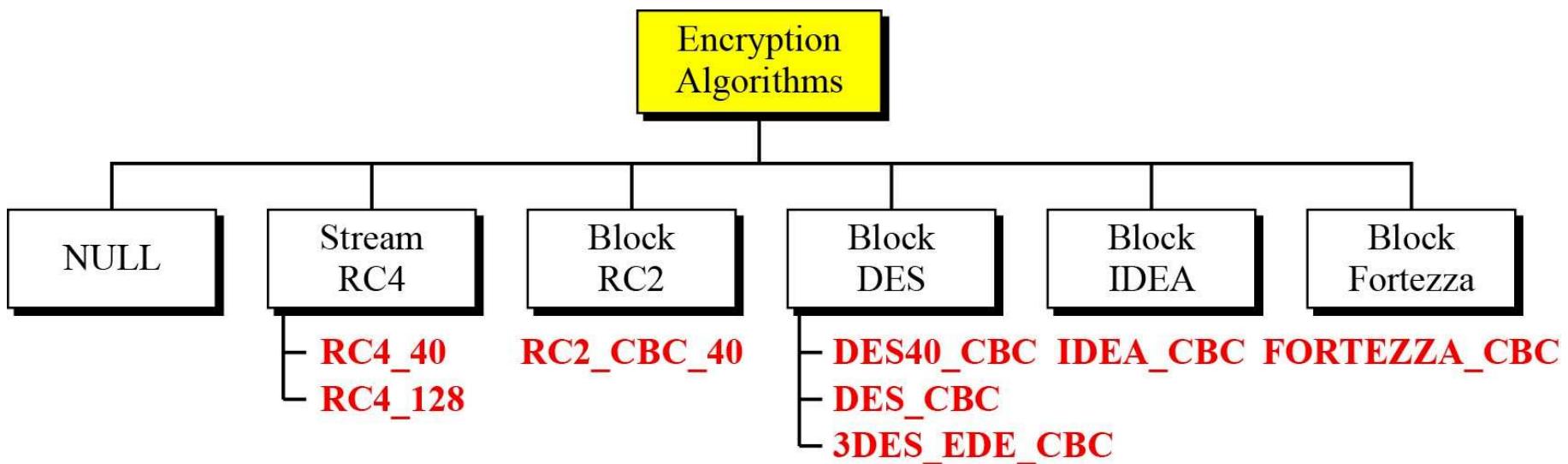
*Another solution is the fixed Diffie-Hellman method. All entities in a group can prepare fixed Diffie-Hellman parameters ( $g$  and  $p$ ).*

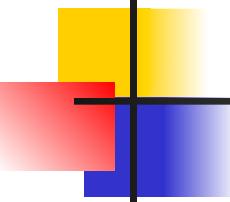
### **Fortezza**

*Fortezza is a registered trademark of the U.S. National Security Agency (NSA). It is a family of security protocols developed for the Defense Department.*

## 17.1.3 Encryption/Decryption Algorithms

Figure 17.6 Encryption/decryption algorithms





## **17.1.3 Continued**

### **NULL**

*The NULL category simply defines the lack of an encryption/decryption algorithm.*

### **Stream RC**

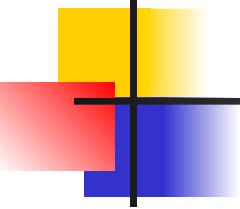
*Two RC algorithms are defined in stream mode.*

### **Block RC**

*One RC algorithm is defined in block mode.*

### **DES**

*All DES algorithms are defined in block mode.*



## **17.1.3 Continued**

### **IDEA**

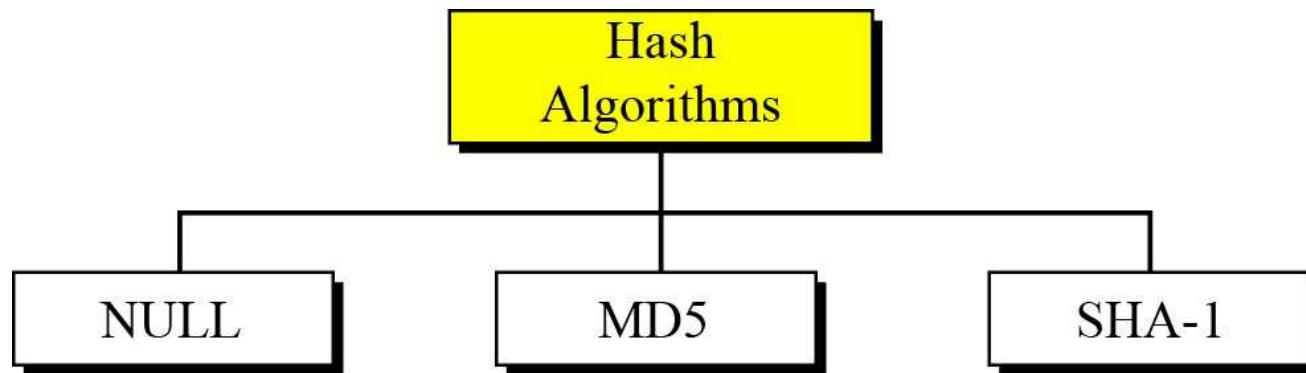
*The IDEA algorithm defined in block mode is IDEA\_CBC, with a 128-bit key.*

### **Fortezza**

*The one Fortezza algorithm defined in block mode is FORTEZZA\_CBC.*

## 17.1.4 Hash Algorithm

Figure 17.7 *Hash algorithms for message integrity*



## **17.1.4 Continued**

### **NULL**

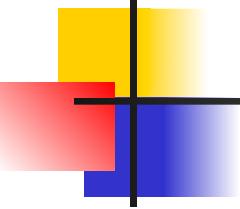
*The two parties may decline to use an algorithm. In this case, there is no hash function and the message is not authenticated.*

### **MD5**

*The two parties may choose MD5 as the hash algorithm. In this case, a 128-key MD5 hash algorithm is used.*

### **SHA-1**

*The two parties may choose SHA as the hash algorithm. In this case, a 160-bit SHA-1 hash algorithm is used.*



## 17.1.5 Cipher Suite

*The combination of key exchange, hash, and encryption algorithms defines a cipher suite for each SSL session.*

```
SSL_DHE_RSA_WITH_DES_CBC_SHA
```

## 17.1.5 Continued

**Table 17.1 SSL cipher suite list**

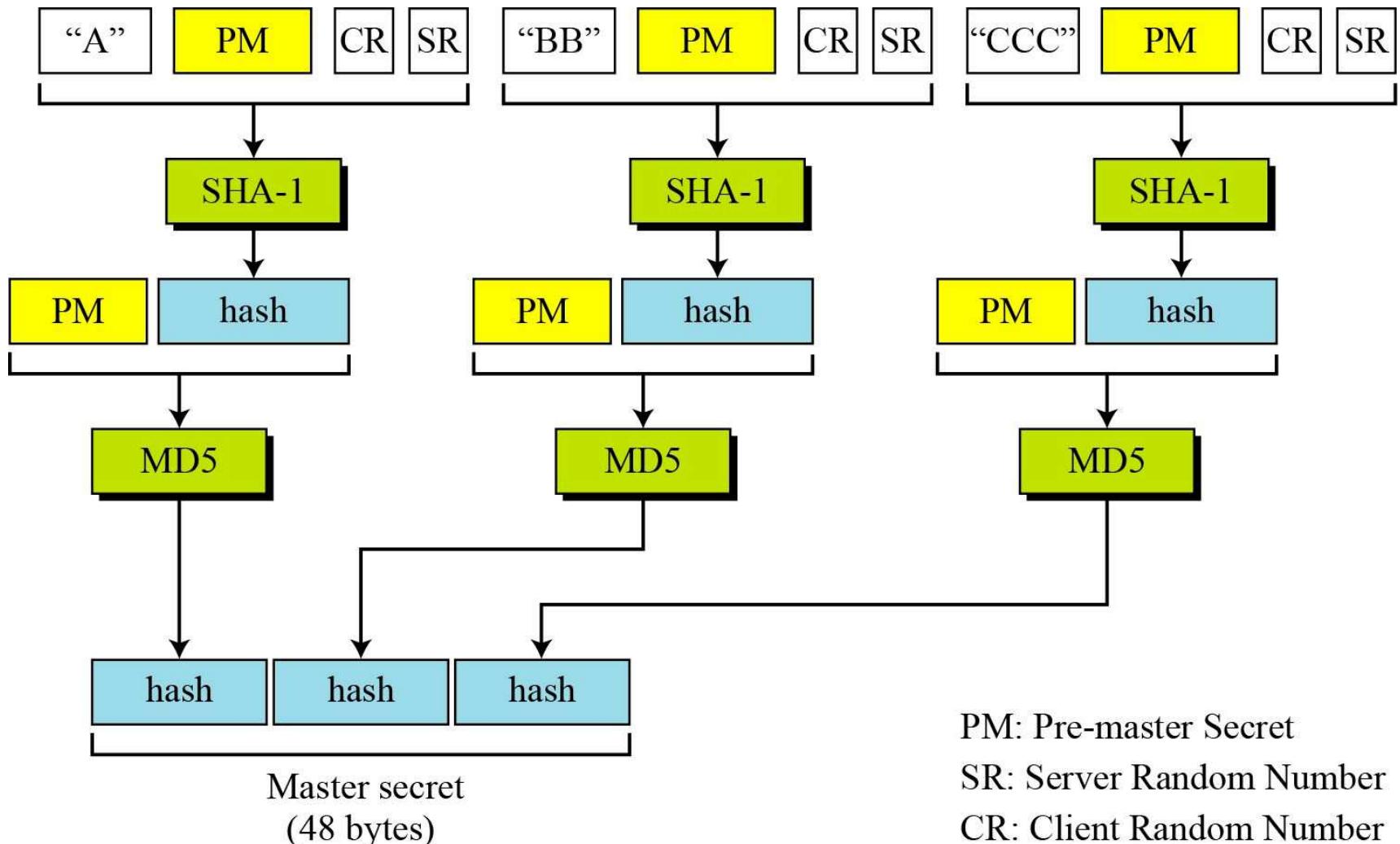
<i>Cipher suite</i>	<i>Key Exchange</i>	<i>Encryption</i>	<i>Hash</i>
SSL_NULL_WITH_NULL_NULL	NULL	NULL	NULL
SSL_RSA_WITH_NULL_MD5	RSA	NULL	MD5
SSL_RSA_WITH_NULL_SHA	RSA	NULL	SHA-1
SSL_RSA_WITH_RC4_128_MD5	RSA	RC4	MD5
SSL_RSA_WITH_RC4_128_SHA	RSA	RC4	SHA-1
SSL_RSA_WITH_IDEA_CBC_SHA	RSA	IDEA	SHA-1
SSL_RSA_WITH DES_CBC_SHA	RSA	DES	SHA-1
SSL_RSA_WITH_3DES_EDE_CBC_SHA	RSA	3DES	SHA-1
SSL_DH_anon_WITH_RC4_128_MD5	DH_anon	RC4	MD5
SSL_DH_anon_WITH DES_CBC_SHA	DH_anon	DES	SHA-1
SSL_DH_anon_WITH_3DES_EDE_CBC_SHA	DH_anon	3DES	SHA-1
SSL_DHE_RSA_WITH DES_CBC_SHA	DHE_RSA	DES	SHA-1
SSL_DHE_RSA_WITH_3DES_EDE_CBC_SHA	DHE_RSA	3DES	SHA-1
SSL_DHE_DSS_WITH DES_CBC_SHA	DHE_DSS	DES	SHA-1
SSL_DHE_DSS_WITH_3DES_EDE_CBC_SHA	DHE_DSS	3DES	SHA-1
SSL_DH_RSA_WITH DES_CBC_SHA	DH_RSA	DES	SHA-1
SSL_DH_RSA_WITH_3DES_EDE_CBC_SHA	DH_RSA	3DES	SHA-1
SSL_DH_DSS_WITH DES_CBC_SHA	DH_DSS	DES	SHA-1
SSL_DH_DSS_WITH_3DES_EDE_CBC_SHA	DH_DSS	3DES	SHA-1
SSL_FORTEZZA_DMS_WITH NULL_SHA	Fortezza	NULL	SHA-1
SSL_FORTEZZA_DMS_WITH_FORTEZZA_CBC_SHA	Fortezza	Fortezza	SHA-1
SSL_FORTEZZA_DMS_WITH_RC4_128_SHA	Fortezza	RC4	SHA-1

## **17.1.6 Compression Algorithms**

*Compression is optional in SSLv3. No specific compression algorithm is defined for SSLv3. Therefore, the default compression method is NULL.*

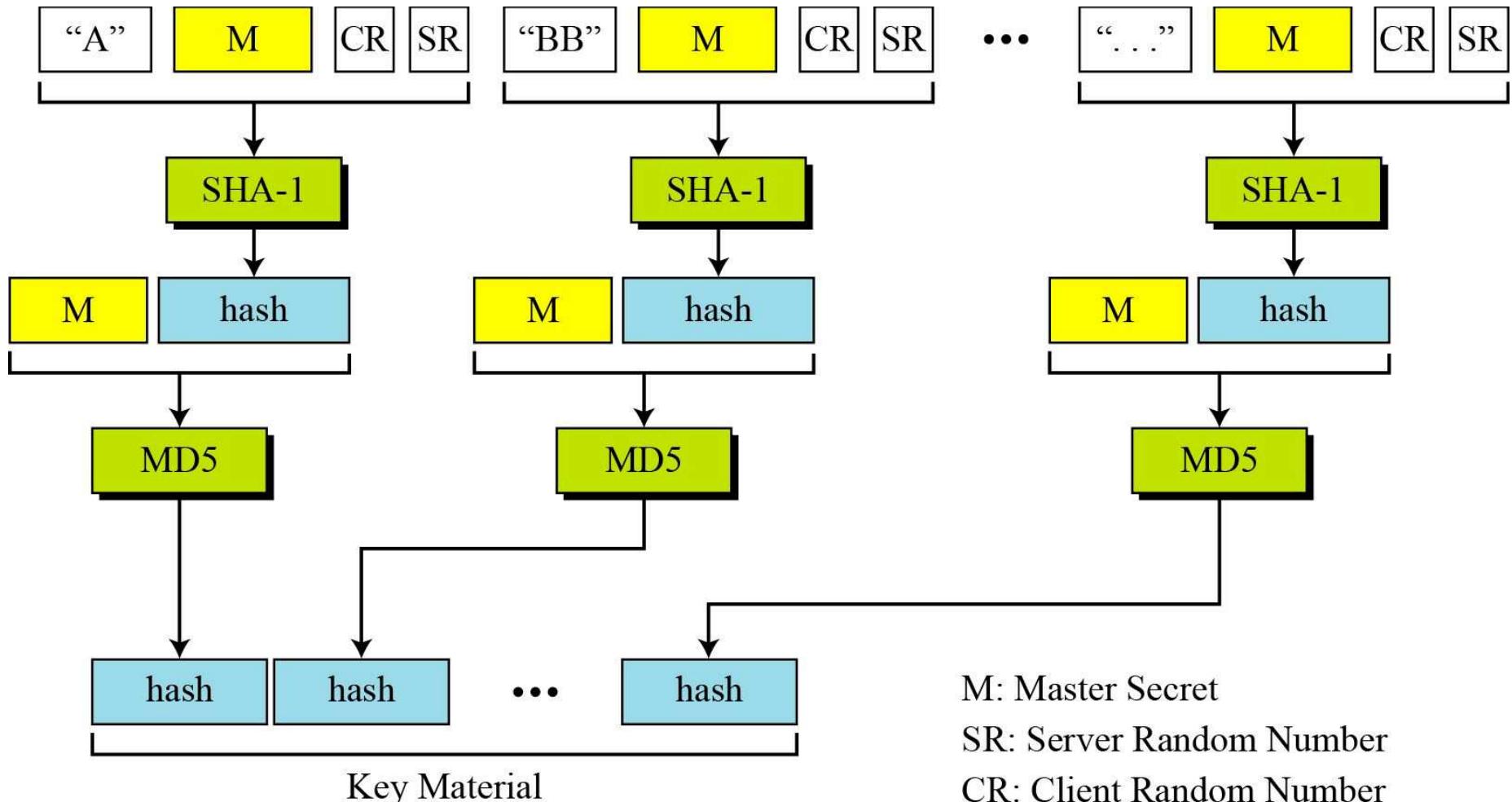
## 17.1.7 Cryptographic Parameter Generation

Figure 17.8 Calculation of master secret from pre-master secret



## 17.1.7 Continued

Figure 17.9 Calculation of key material from master secret



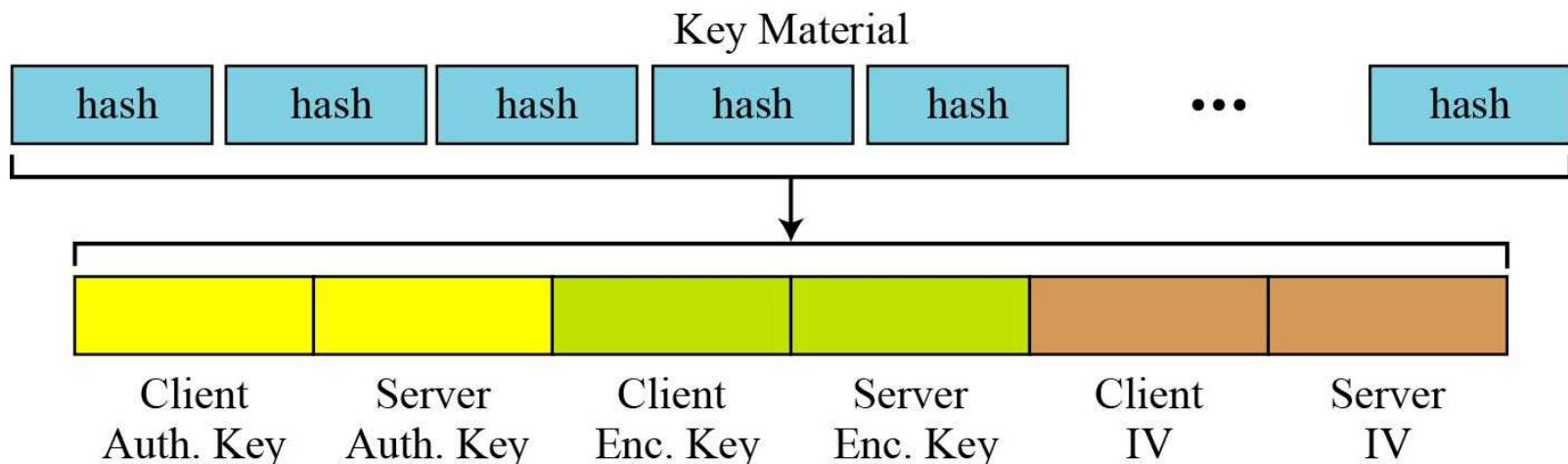
## 17.1.7 *Continued*

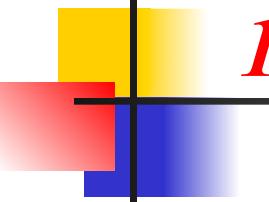
**Figure 17.10** *Extractions of cryptographic secrets from key material*

Auth. Key: Authentication Key

Enc. Key: Encryption Key

IV: Initialization Vector





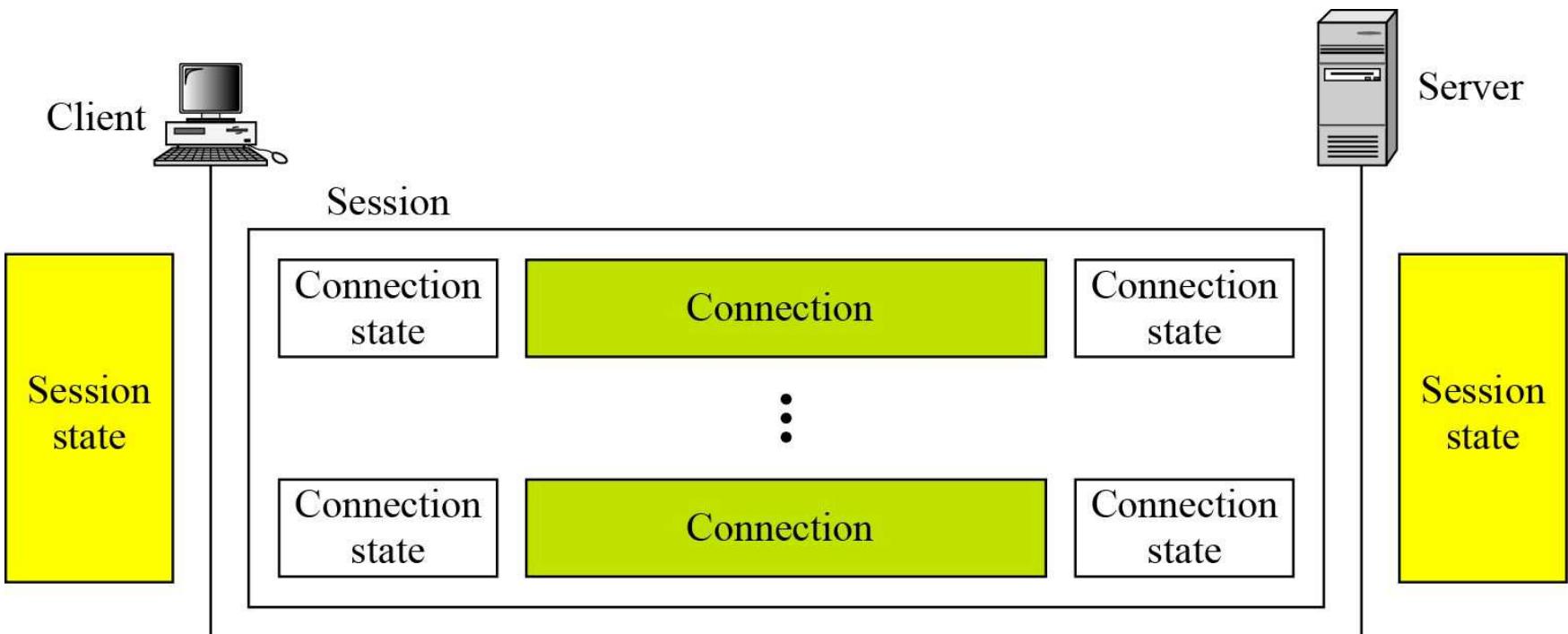
## *17.1.8 Sessions and Connections*

**Note**

**In a session, one party has the role of a client  
and the other the role of a server;  
in a connection, both parties have equal  
roles, they are peers.**

## 17.1.8 *Continued*

**Figure 17.11** *A session and connections*



## 17.1.8 *Continued*

### Session State

**Table 17.2 *Session state parameters***

<i>Parameter</i>	<i>Description</i>
Session ID	A server-chosen 8-bit number defining a session.
Peer Certificate	A certificate of type X509.v3. This parameter may by empty (null).
Compression Method	The compression method.
Cipher Suite	The agreed-upon cipher suite.
Master Secret	The 48-byte secret.
Is resumable	A yes-no flag that allows new connections in an old session.

## 17.1.8 *Continued*

### Connection State

**Table 17.3** *Connection state parameters*

<i>Parameter</i>	<i>Description</i>
Server and client random numbers	A sequence of bytes chosen by the server and client for each connection.
Server write MAC secret	The outbound server MAC key for message integrity. The server uses it to sign; the client uses it to verify.
Client write MAC secret	The outbound client MAC key for message integrity. The client uses it to sign; the server uses it to verify.
Server write secret	The outbound server encryption key for message integrity.
Client write secret	The outbound client encryption key for message integrity.
Initialization vectors	The block ciphers in CBC mode use initialization vectors (IVs). One initialization vector is defined for each cipher key during the negotiation, which is used for the first block exchange. The final cipher text from a block is used as the IV for the next block.
Sequence numbers	Each party has a sequence number. The sequence number starts from 0 and increments. It must not exceed $2^{64} - 1$ .

## **17.1.8 *Continued***

### **Note**

**The client and the server have six different cryptography secrets: three read secrets and three write secrets.**

**The read secrets for the client are the same as the write secrets for the server and vice versa.**

## 17-2 Four Protocols

*We have discussed the idea of SSL without showing how SSL accomplishes its tasks. SSL defines four protocols in two layers, as shown in Figure 17.12.*

### Topics discussed in this section:

17.2.1 Handshake Protocol

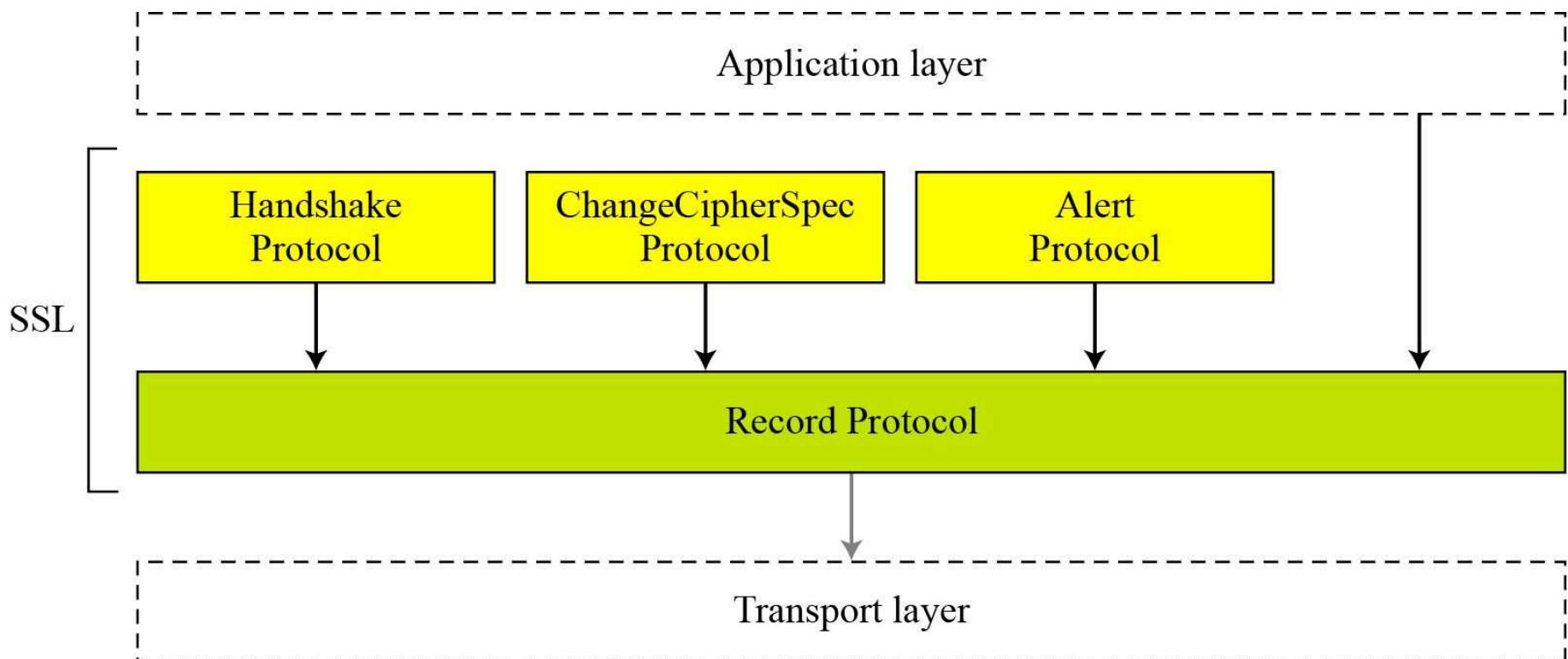
17.2.2 ChangeCipher Spec Protocol

17.2.3 Alert Protocol

17.2.4 Record Protocol

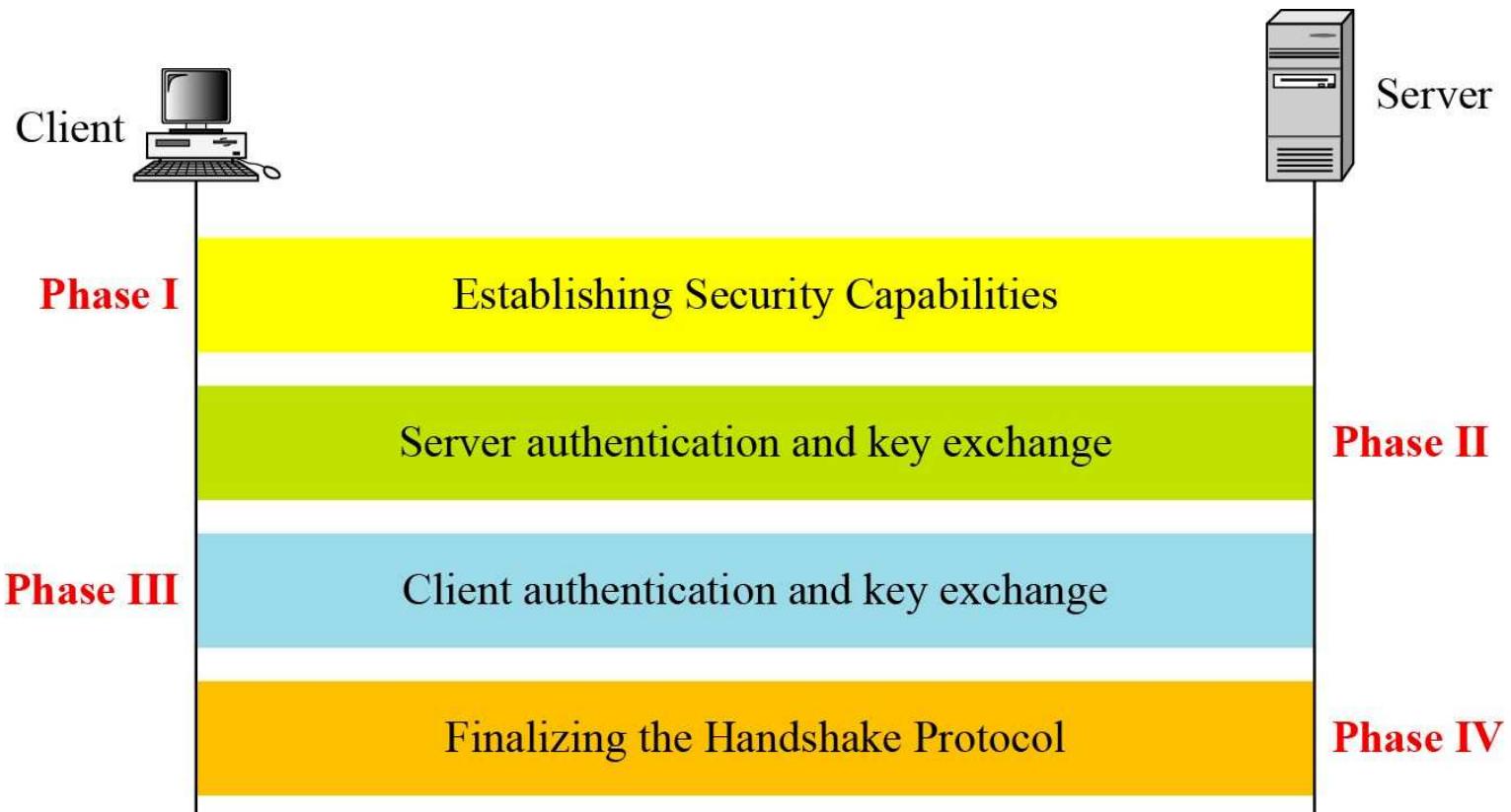
## 17.2. Continued

Figure 17.12 Four SSL protocols



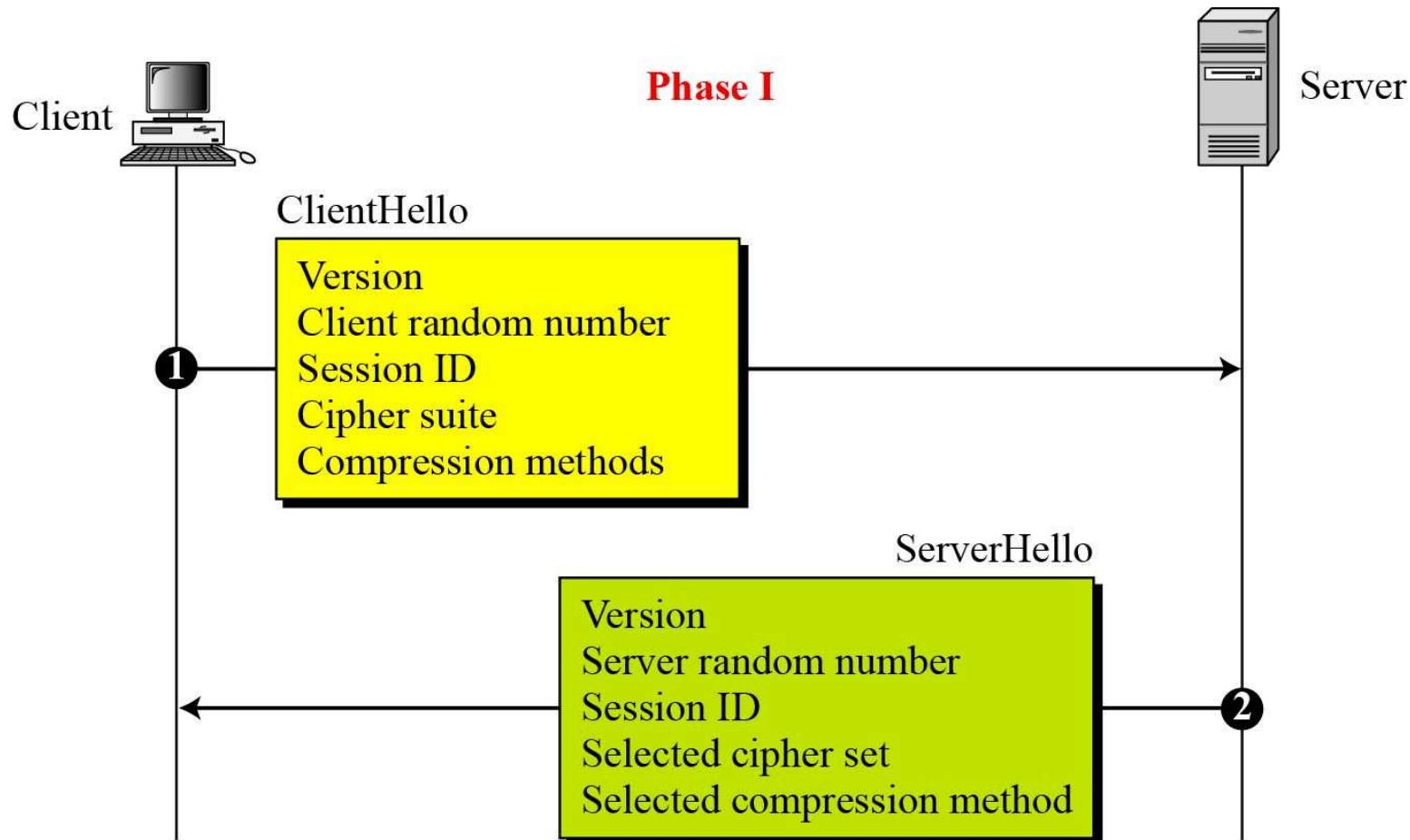
## 17.2.1 Handshake Protocol

Figure 17.13 Handshake Protocol



## 17.2.1 Continued

Figure 17.14 Phase I of Handshake Protocol



## **17.2.1 *Continued***

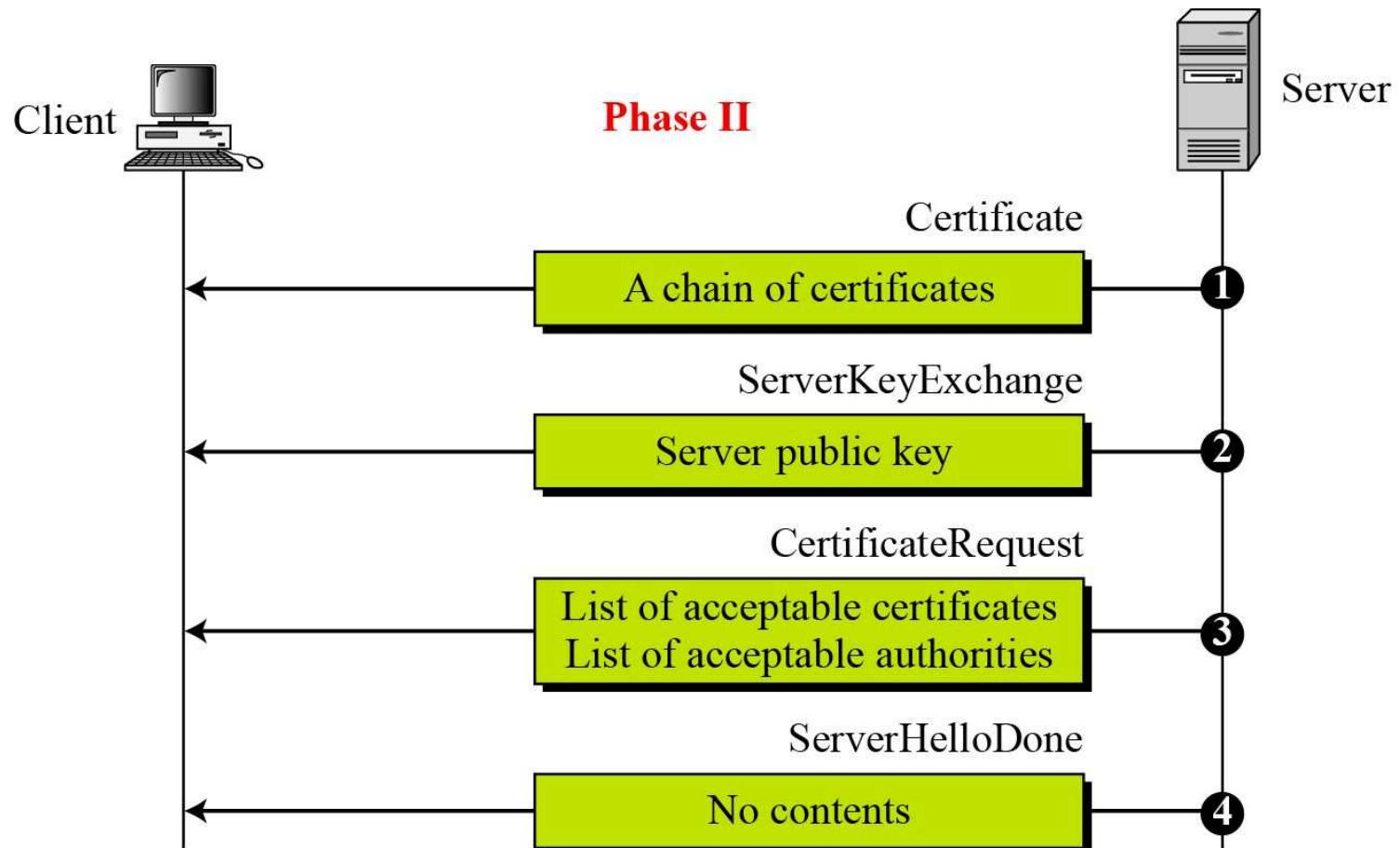
### **Note**

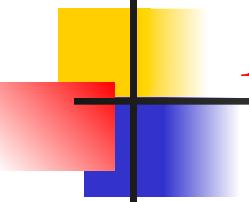
**After Phase I, the client and server know the following:**

- ❑ The version of SSL**
- ❑ The algorithms for key exchange, message authentication, and encryption**
- ❑ The compression method**
- ❑ The two random numbers for key generation**

## 17.2.1 Continued

Figure 17.15 Phase II of Handshake Protocol





## **17.2.1 *Continued***

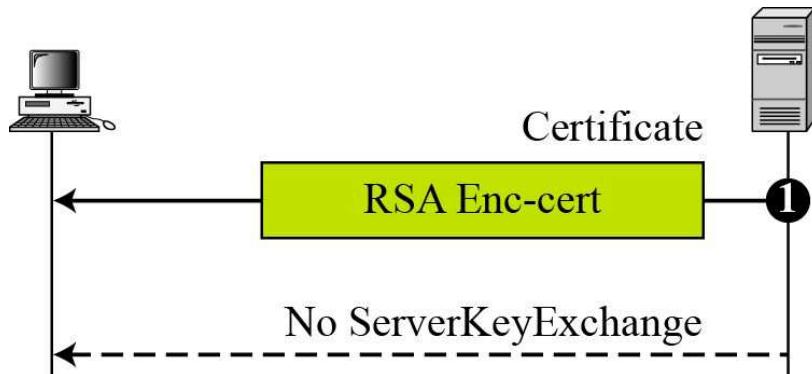
***Note***

**After Phase II,**

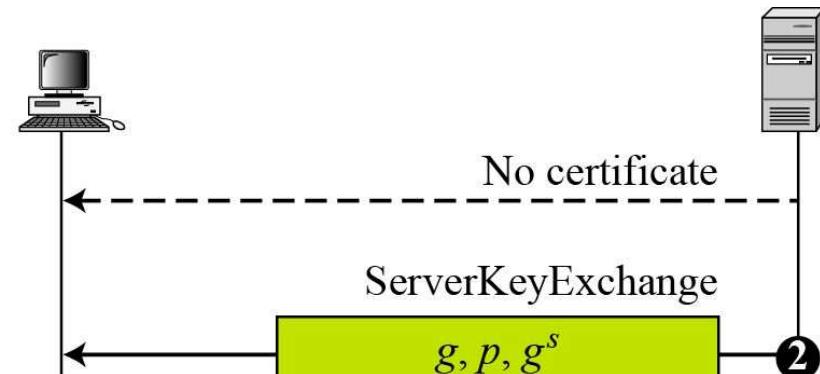
- The server is authenticated to the client.
- The client knows the public key of the server if required.

## 17.2.1 Continued

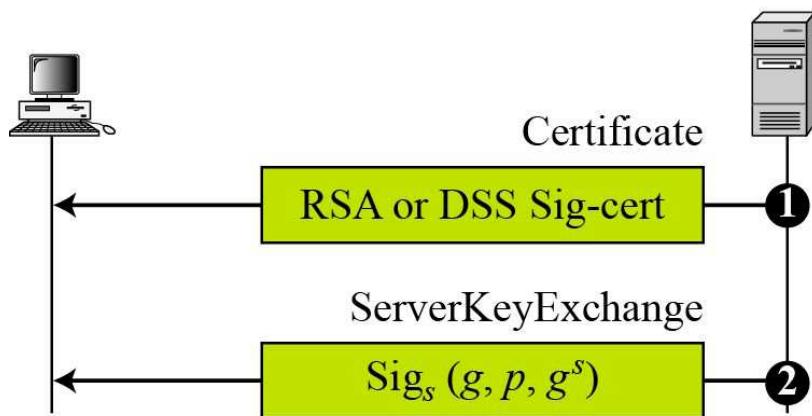
Figure 17.16 Four cases in Phase II



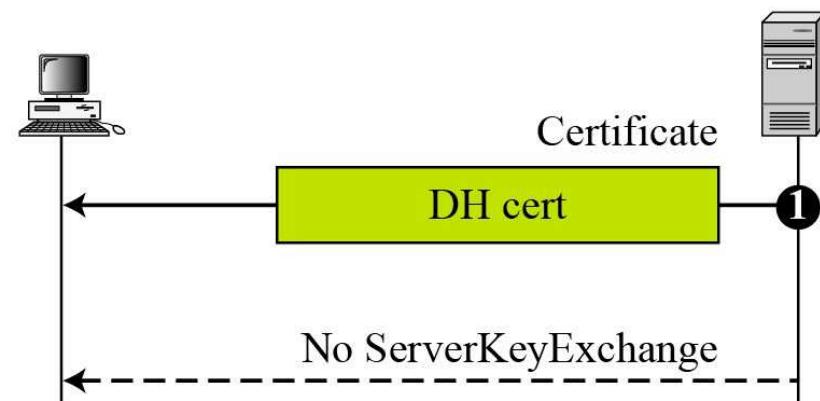
a. RSA



b. Anonymous DH



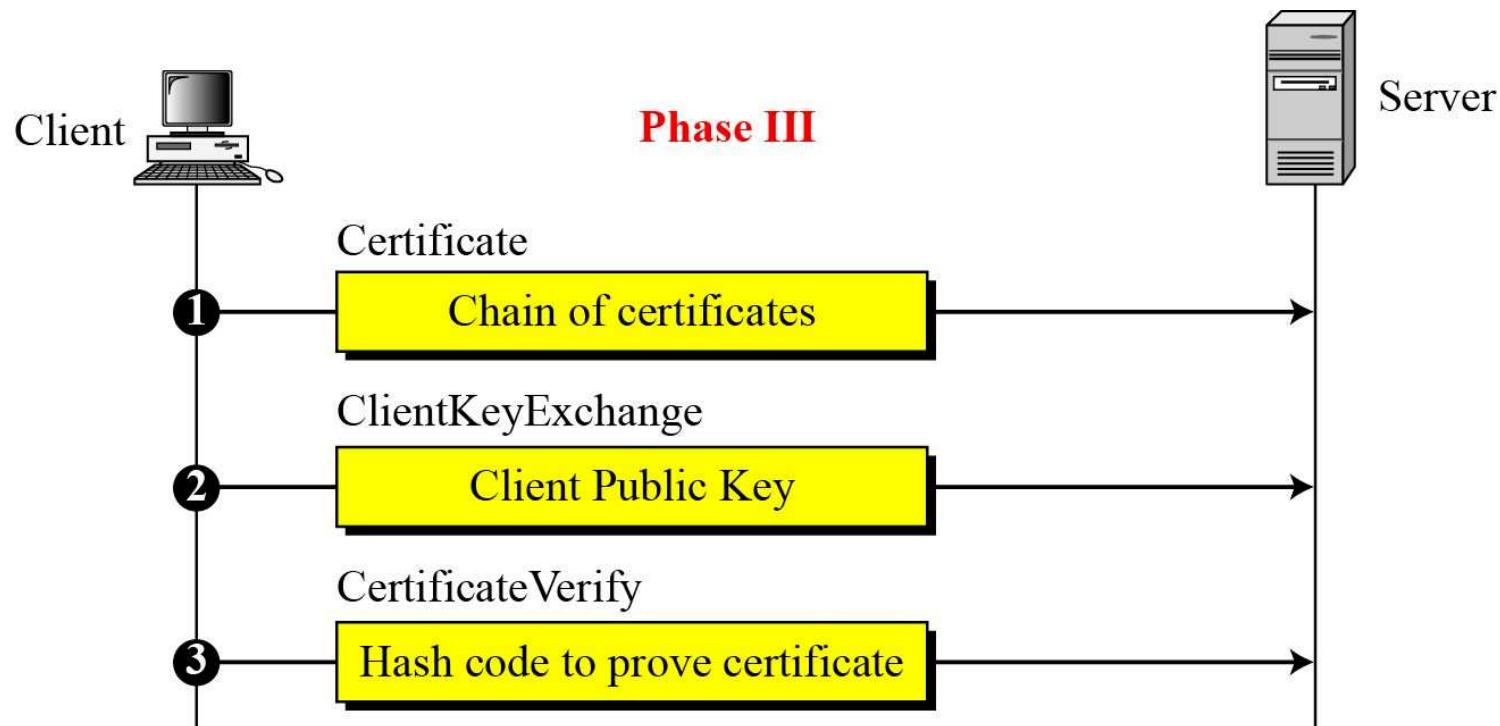
c. Ephemeral DH

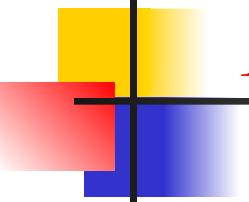


d. Fixed DH

## 17.2.1 *Continued*

**Figure 17.17 Phase III of Handshake Protocol**





## **17.2.1 *Continued***

***Note***

**After Phase III,**

- The client is authenticated for the server.**
- Both the client and the server know the pre-master secret.**

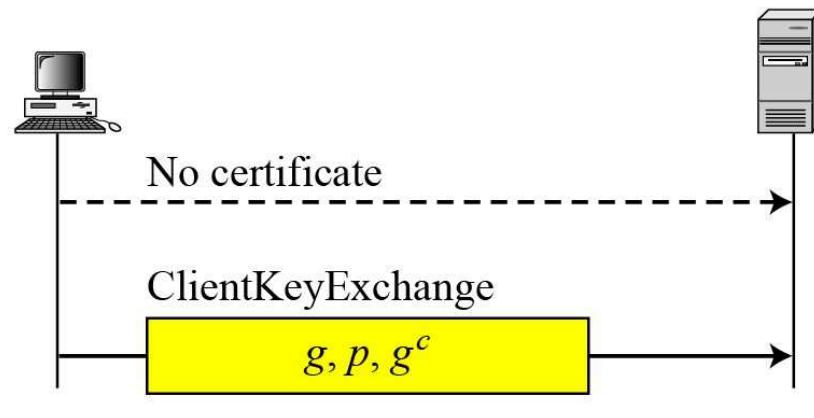
## 17.2.1 Continued

Figure 17.18 Four cases in Phase III

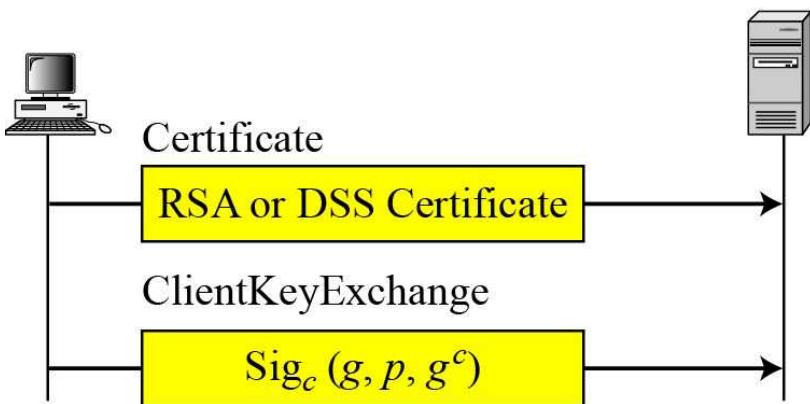
S encrypted with server's public key  
Sig<sub>c</sub>: Signed with client's public key



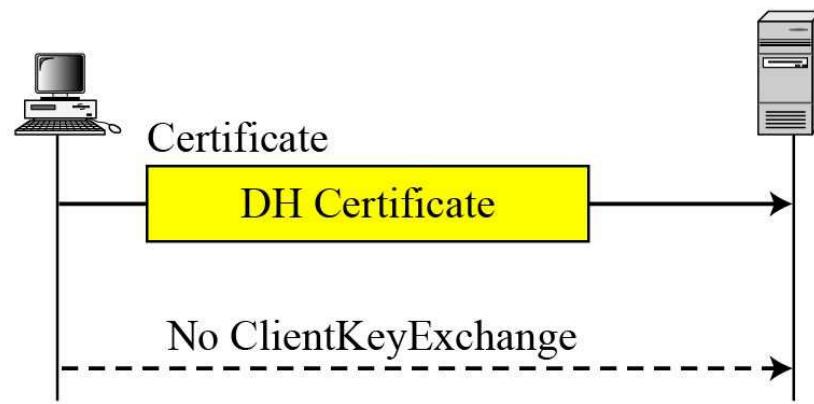
a. RSA



b. Anonymous DH



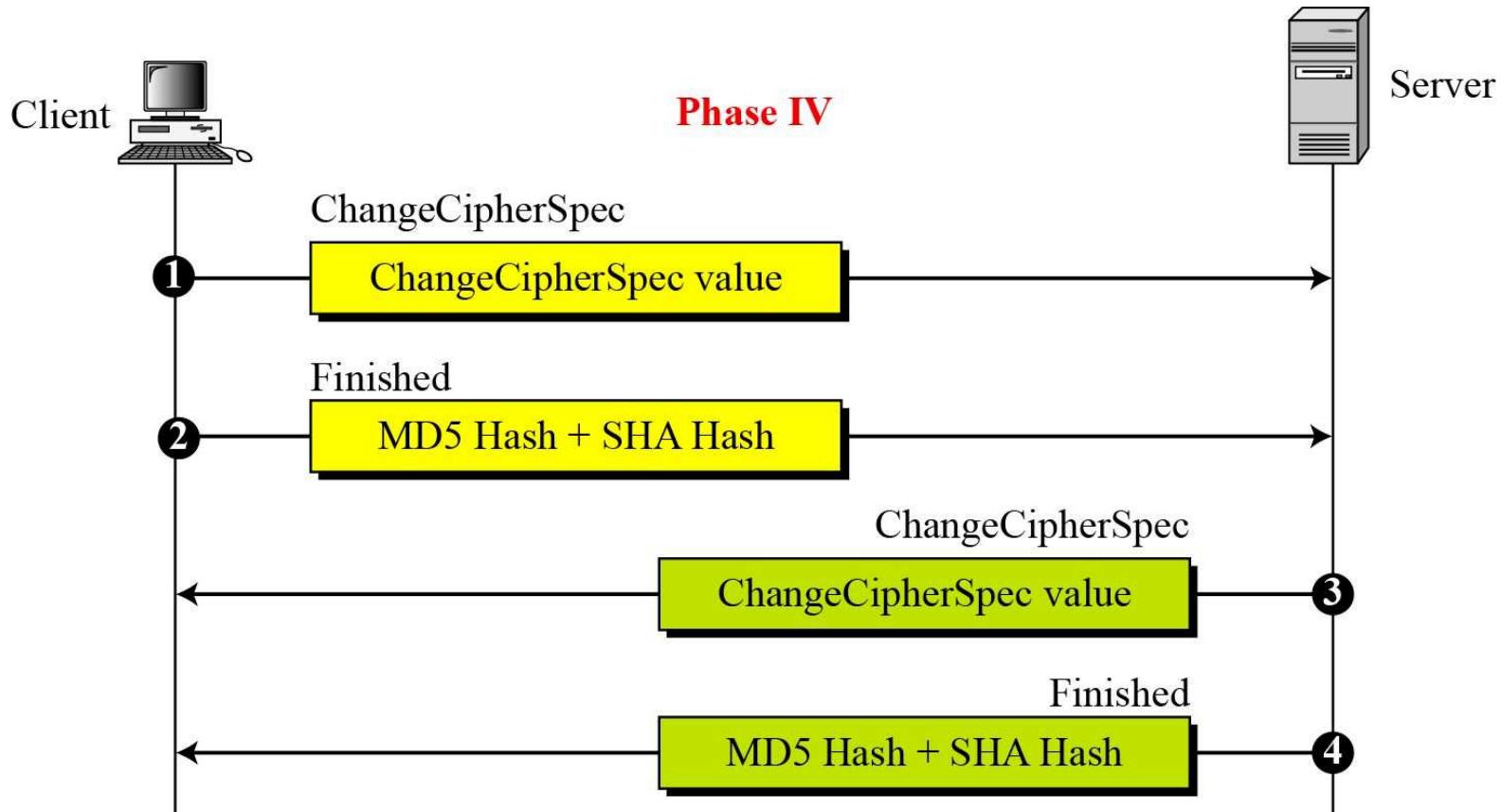
c. Ephemeral DH

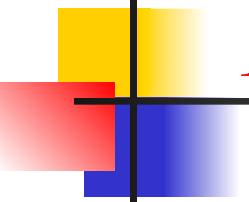


d. Fixed DH

## 17.2.1 Continued

Figure 17.19 Phase IV of Handshake Protocol





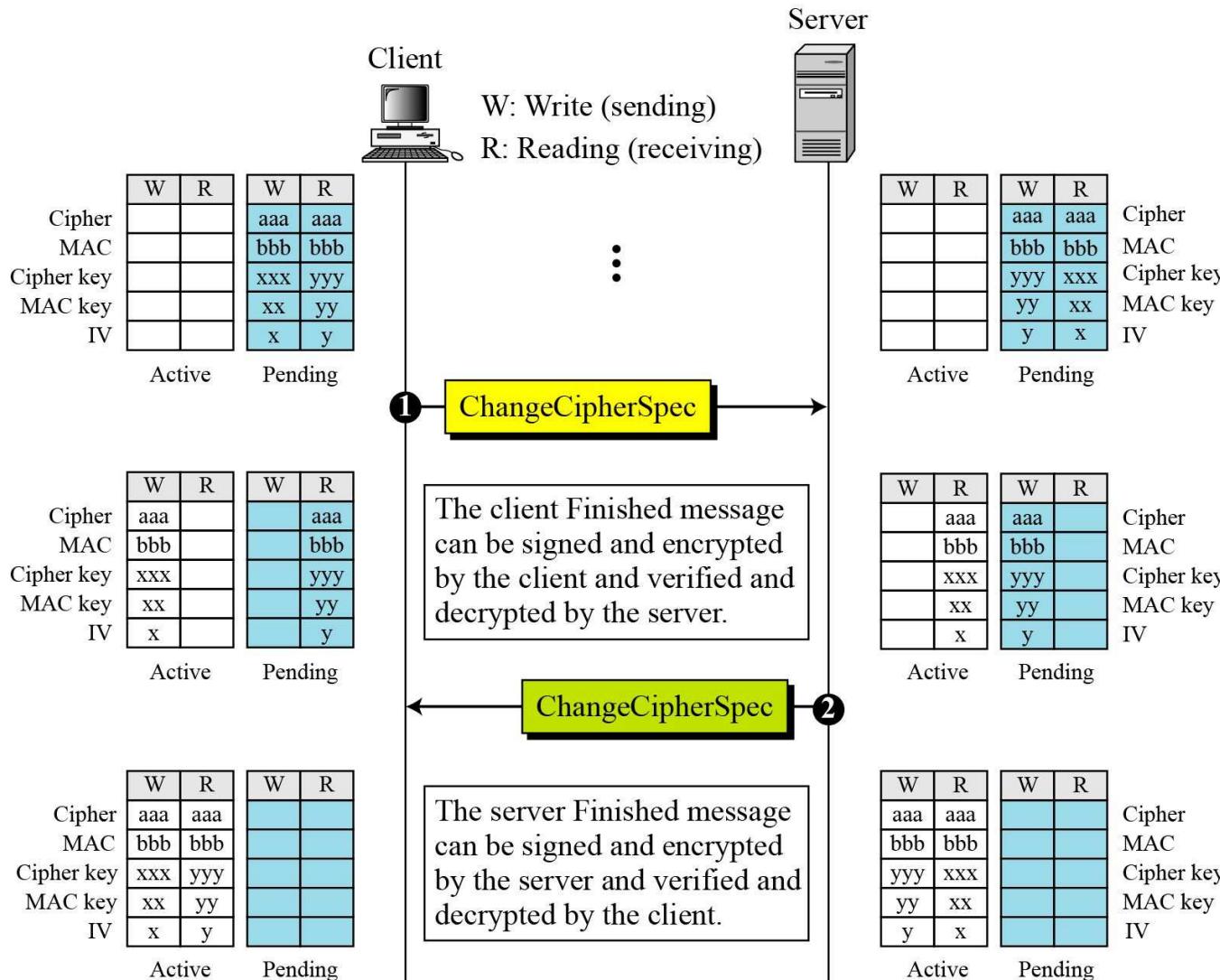
## **17.2.1 *Continued***

***Note***

**After Phase IV, the client and server are ready to exchange data.**

## 17.2.2 ChangeCipherSpec Protocol

Figure 17.20 Movement of parameters from pending state to active state



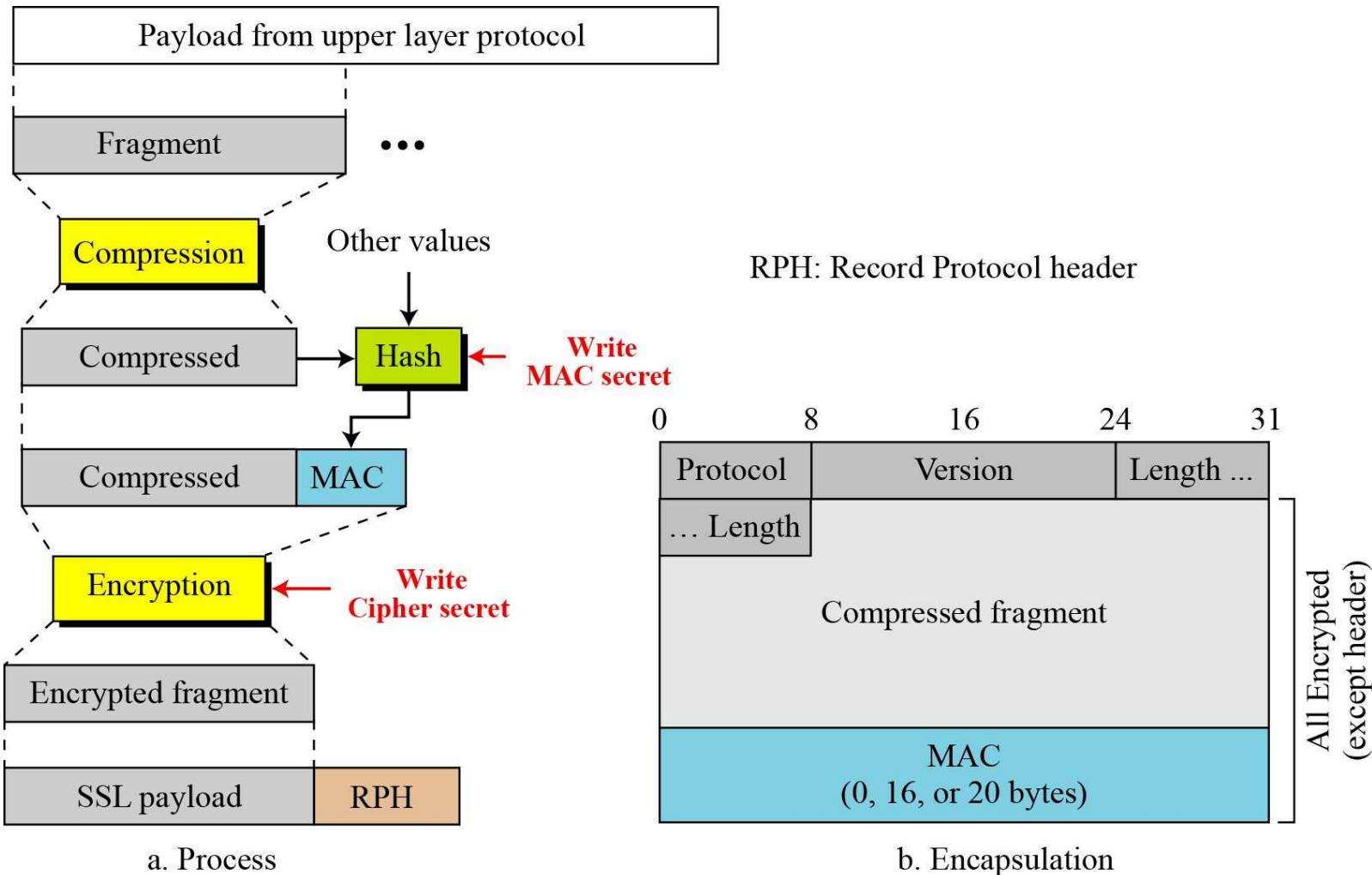
## 17.2.3 Alert Protocol

**Table 17.4 Alerts defined for SSL**

<i>Value</i>	<i>Description</i>	<i>Meaning</i>
0	<i>CloseNotify</i>	Sender will not send any more messages.
10	<i>UnexpectedMessage</i>	An inappropriate message received.
20	<i>BadRecordMAC</i>	An incorrect MAC received.
30	<i>DecompressionFailure</i>	Unable to decompress appropriately.
40	<i>HandshakeFailure</i>	Sender unable to finalize the handshake.
41	<i>NoCertificate</i>	Client has no certificate to send.
42	<i>BadCertificate</i>	Received certificate corrupted.
43	<i>UnsupportedCertificate</i>	Type of received certificate is not supported.
44	<i>CertificateRevoked</i>	Signer has revoked the certificate.
45	<i>CertificateExpired</i>	Certificate expired.
46	<i>CertificateUnknown</i>	Certificate unknown.
47	<i>IllegalParameter</i>	An out-of-range or inconsistent field.

## 17.2.4 Record Protocol

Figure 17.21 Processing done by the Record Protocol



a. Process

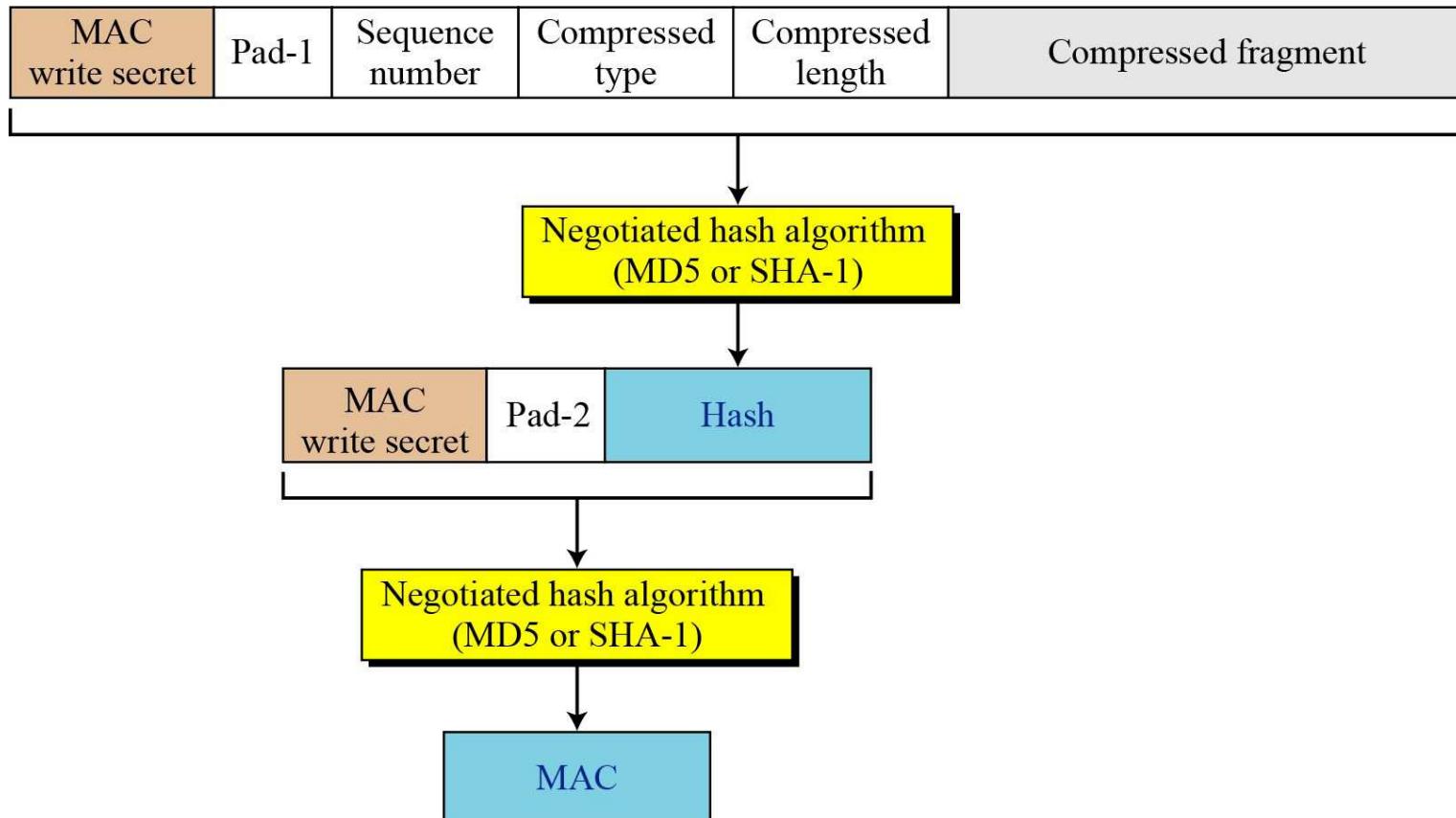
b. Encapsulation

## 17.2.4 Continued

**Figure 17.22 Calculation of MAC**

Pad-1: Byte 0x36 (00110110) repeated 48 times for MD5 and 40 times for SHA-1

Pad-2: Byte 0x5C (01011100) repeated 48 times for MD5 and 40 times for SHA-1



## 17-3 SSL MESSAGE FORMATS

*As we have discussed, messages from three protocols and data from the application layer are encapsulated in the Record Protocol messages.*

**Topics discussed in this section:**

**17.3.1 ChangeCipherSpec Protocol**

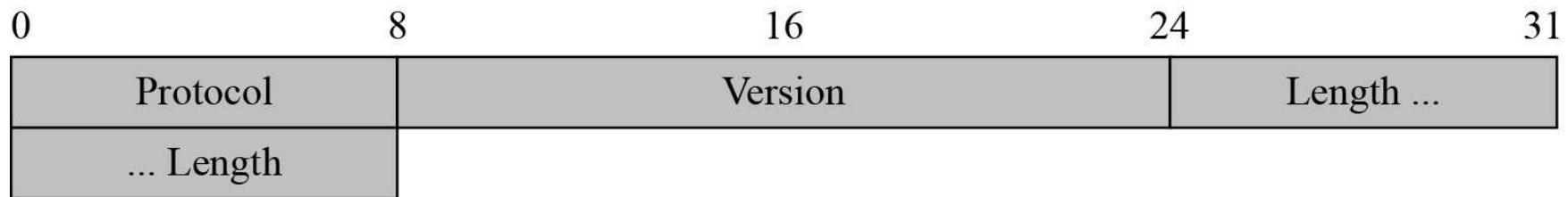
**17.3.2 Alert Protocol**

**17.3.3 Handshake Protocol**

**17.3.4 Application Data**

## 17.3 Continued

**Figure 17.23 Record Protocol general header**



## 17.3.1 *ChangeCipherSpec Protocol*

**Figure 17.24** *ChangeCipherSpec message*

0	8	16	24	31
Protocol: 20		Version		Length: 0
... Length: 1		CCS: 1		

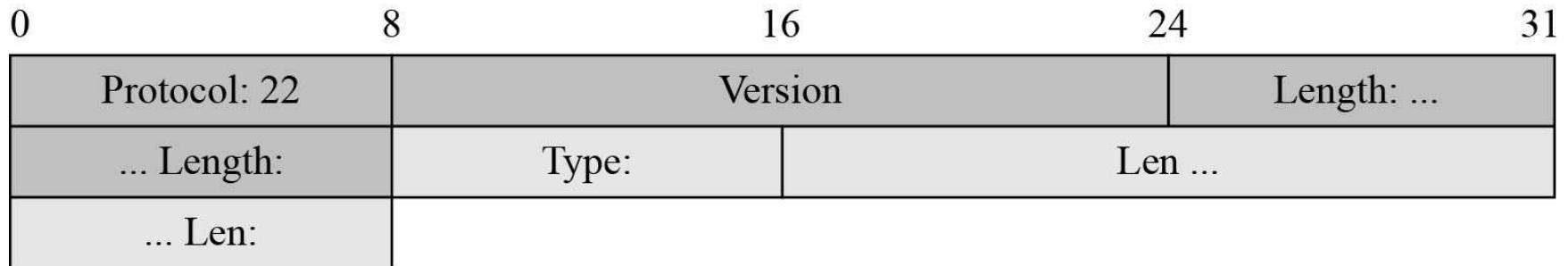
## 17.3.2 Alert Protocol

**Figure 17.25 Alert message**

0	8	16	24	31
Protocol: 21		Version		Length: 0
... Length: 2	Level	Description		

### 17.3.3 Handshake Protocol

**Figure 17.26 Generic header for Handshake Protocol**



### 17.3.3 *Continued*

**Table 17.5** *Types of Handshake messages*

Type	Message
0	HelloRequest
1	ClientHello
2	ServerHello
11	Certificate
12	ServerKeyExchange
13	CertificateRequest
14	ServerHelloDone
15	CertificateVerify
16	ClientKeyExchange
20	Finished

### 17.3.3 Continued

Figure 17.27 Virtual tributary types

0	8	16	24	31
Protocol: 22		Version	Length ...	
... Length: 4	Type: 0		Len ...	
... Len: 0				

### 17.3.3 Continued

Figure 17.28 *ClientHello message*

0	8	16	24	31		
Protocol: 22	Version		Length ...			
... Length	Type: 1	Len ...				
... Len	Proposed version					
Client random number (32 bytes)				ID length		
Session ID (variable length)						
Cipher suite length	Cipher suites (variable numbers, each of 2 bytes)					
Com. methods length	Compression methods (variable number, each of 1 byte)					

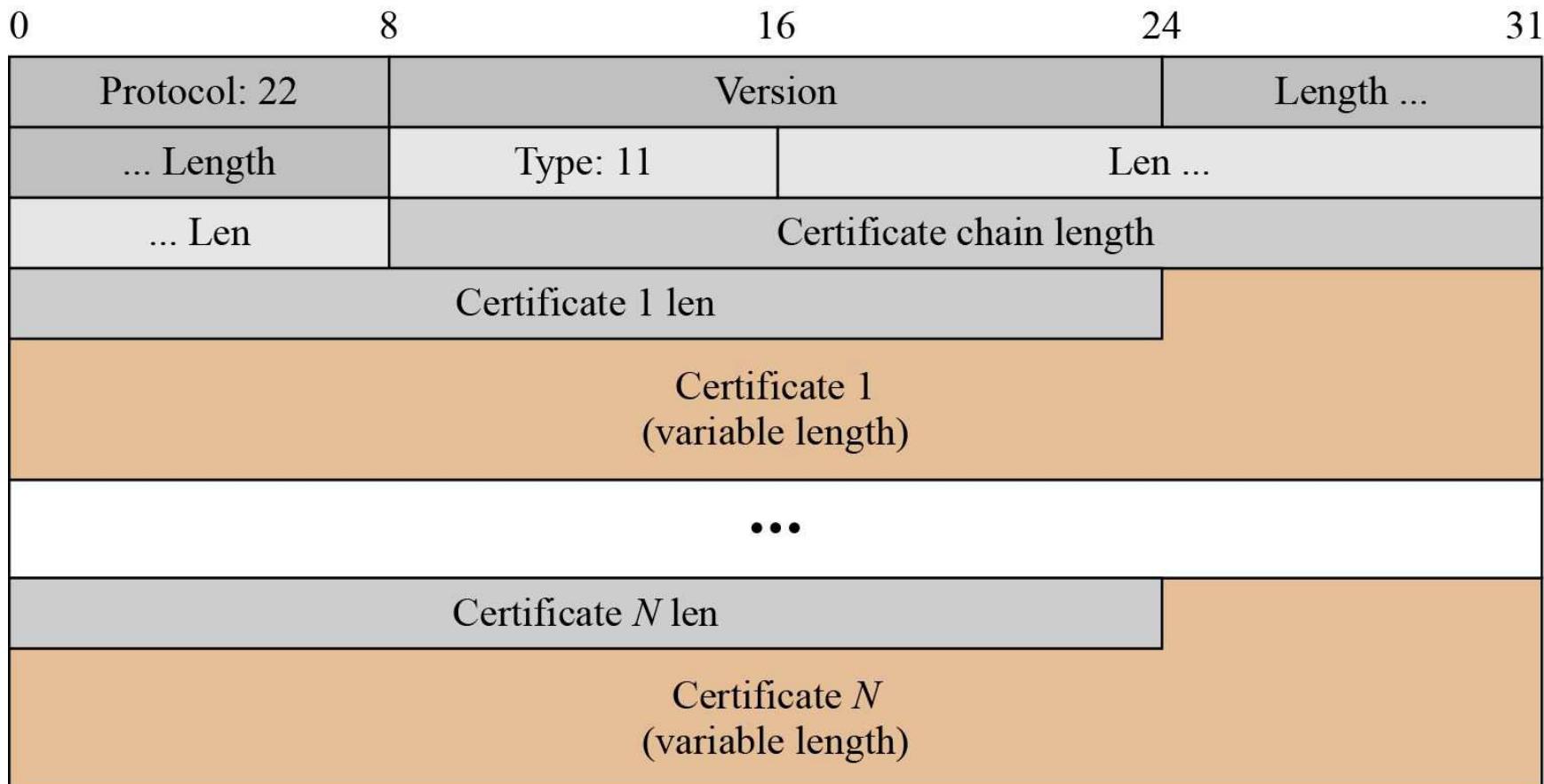
### 17.3.3 Continued

Figure 17.29 *ServerHello message*

0	8	16	24	31		
Protocol: 22	Version		Length ...			
... Length	Type: 2	Len ...				
... Len	Proposed version					
Server random number (32 bytes)			ID length			
Session ID (variable length)						
Selected cipher suite		Selected com.				

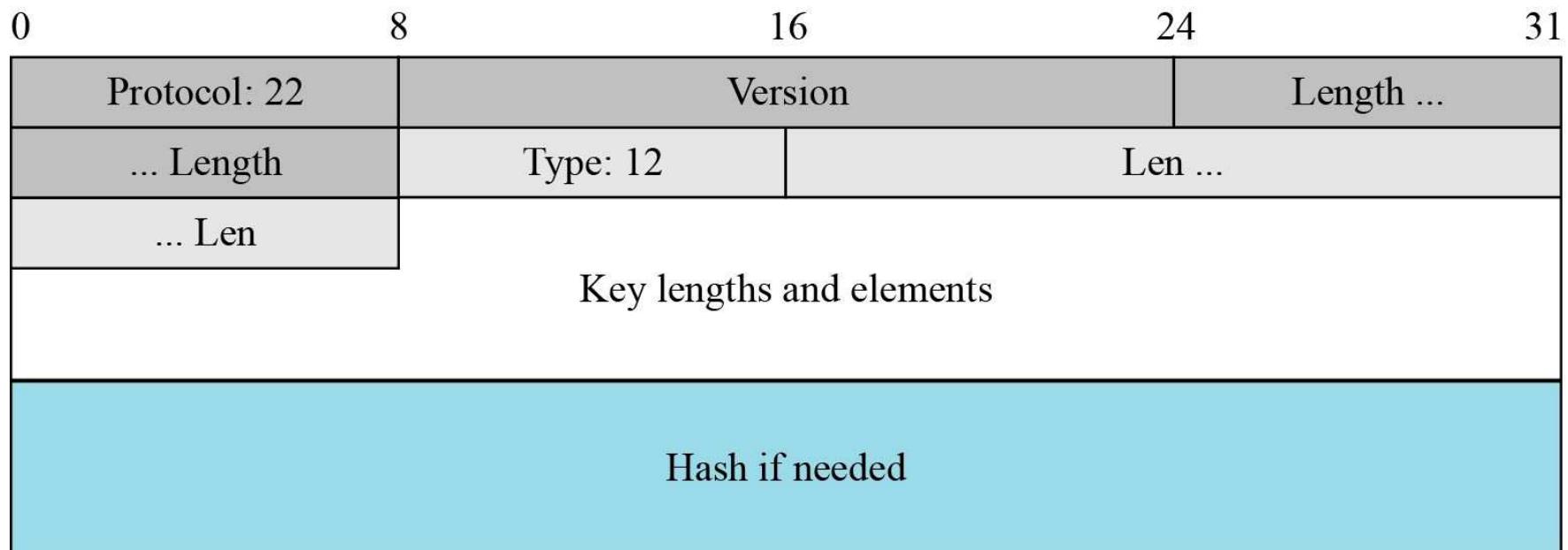
### 17.3.3 Continued

Figure 17.30 Certificate message



### 17.3.3 Continued

Figure 17.31 *ServerKeyExchange message*



### 17.3.3 Continued

Figure 17.32 CertificateRequest message

0	8	16	24	31						
Protocol: 22	Version		Length ...							
... Length	Type: 13	Len ...								
... Len	Len of cert types									
Certificate types (variable number, each of one byte)			Length of CAs							
Length of CA 1 Name										
CA 1 Name										
...										
Length of CA N Name										
CA N Name										

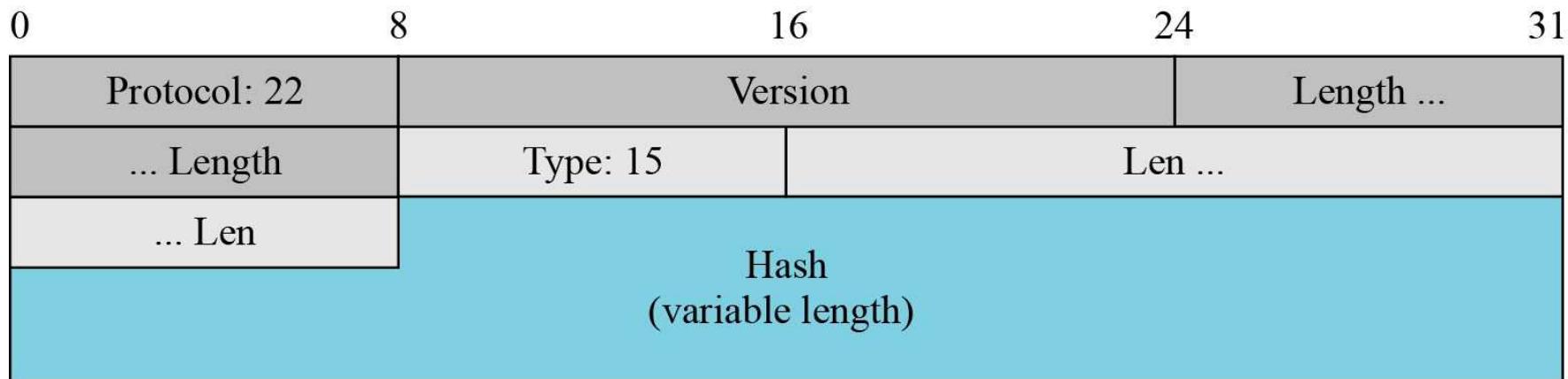
### 17.3.3 Continued

Figure 17.33 *ServerHelloDone message*

0	8	16	24	31		
Protocol: 22	Version		Length ...			
... Length: 4	Type: 14	Len ...				
... Len: 0						

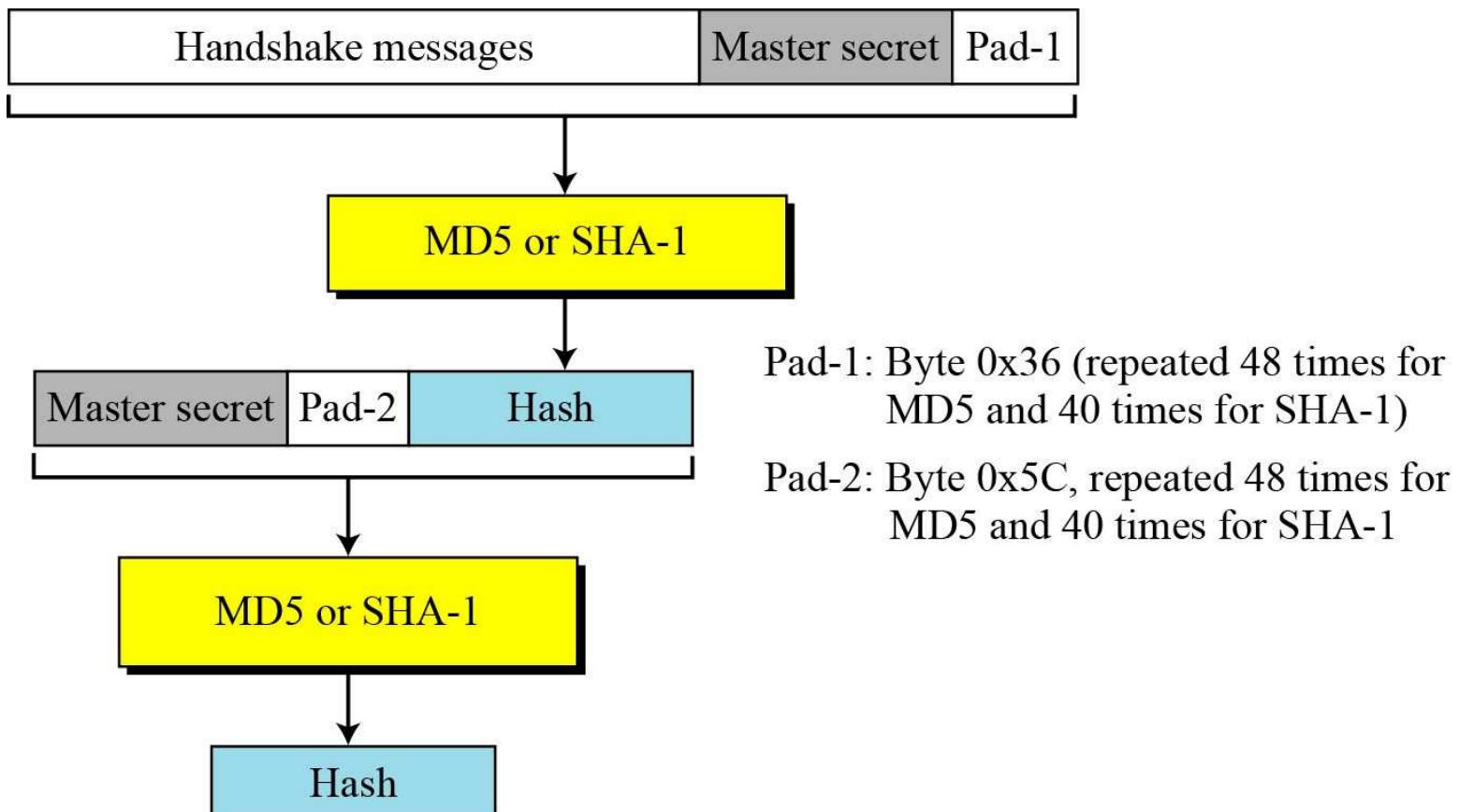
### 17.3.3 Continued

Figure 17.34 CertificateVerify message



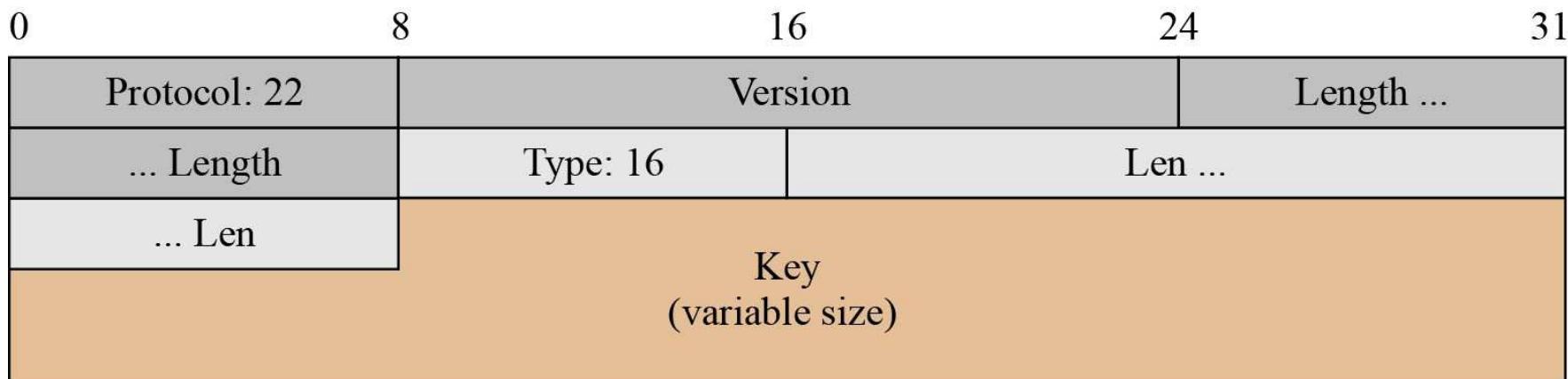
### 17.3.3 Continued

Figure 17.35 Hash calculation for CertificateVerify message



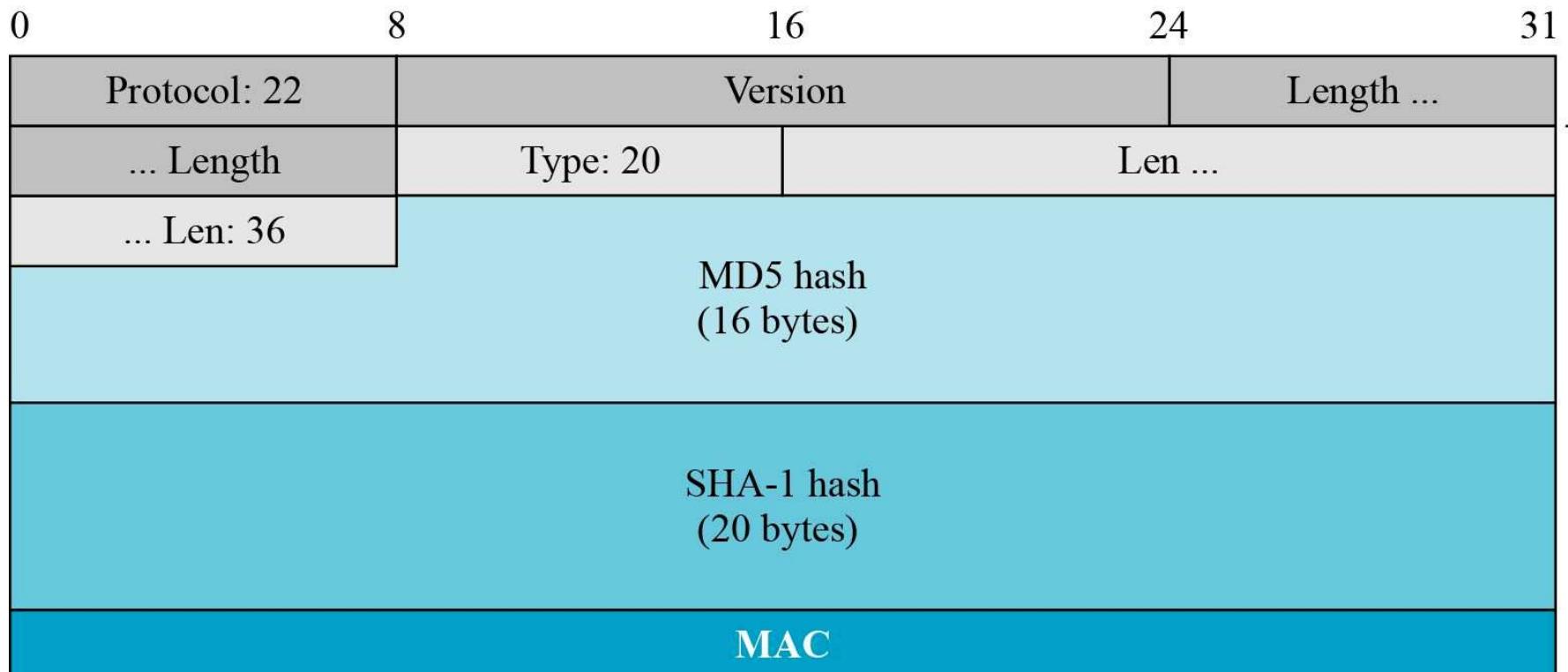
### 17.3.3 Continued

Figure 17.36 *ClientKeyExchange message*



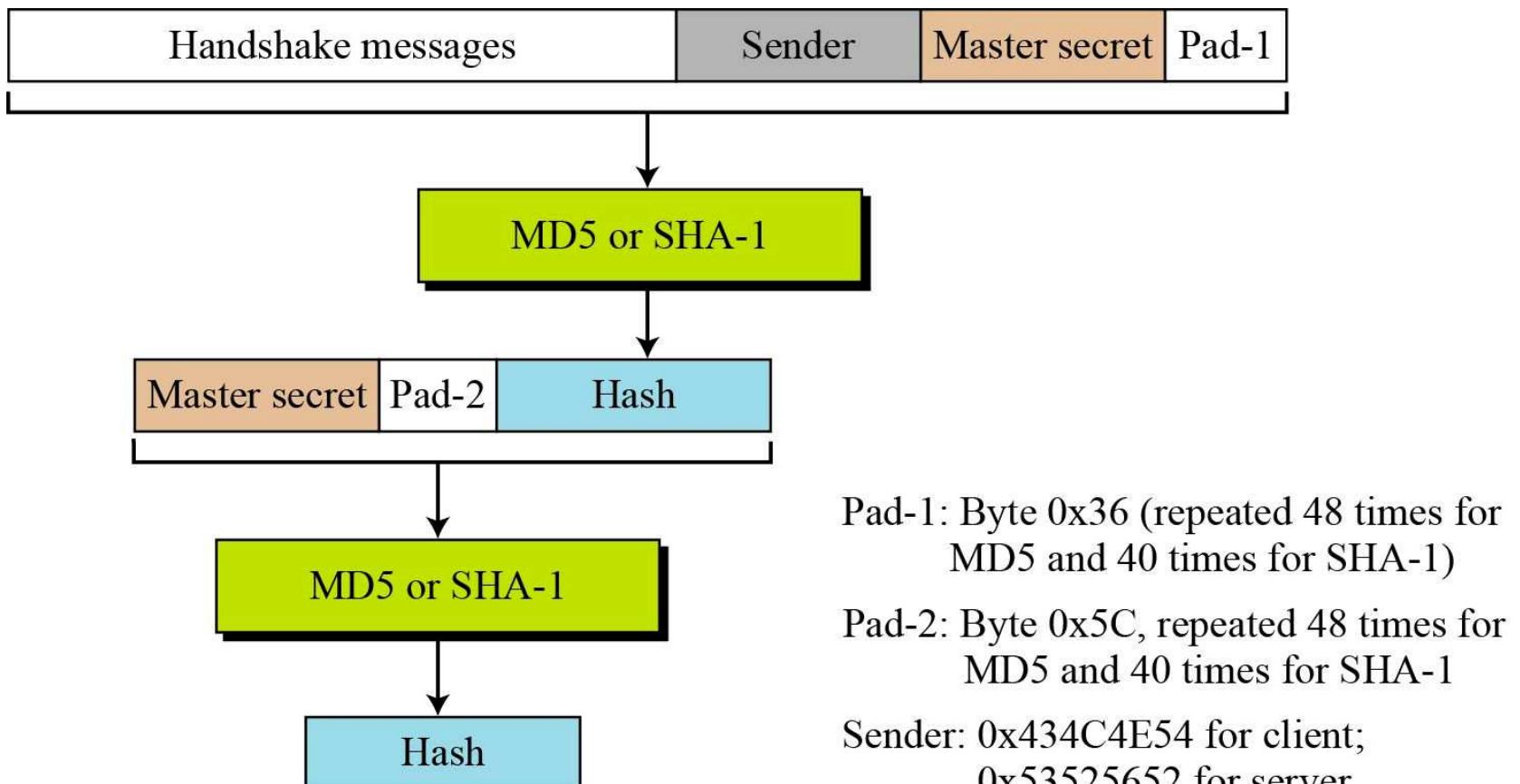
### 17.3.3 Continued

Figure 17.37 Finished message



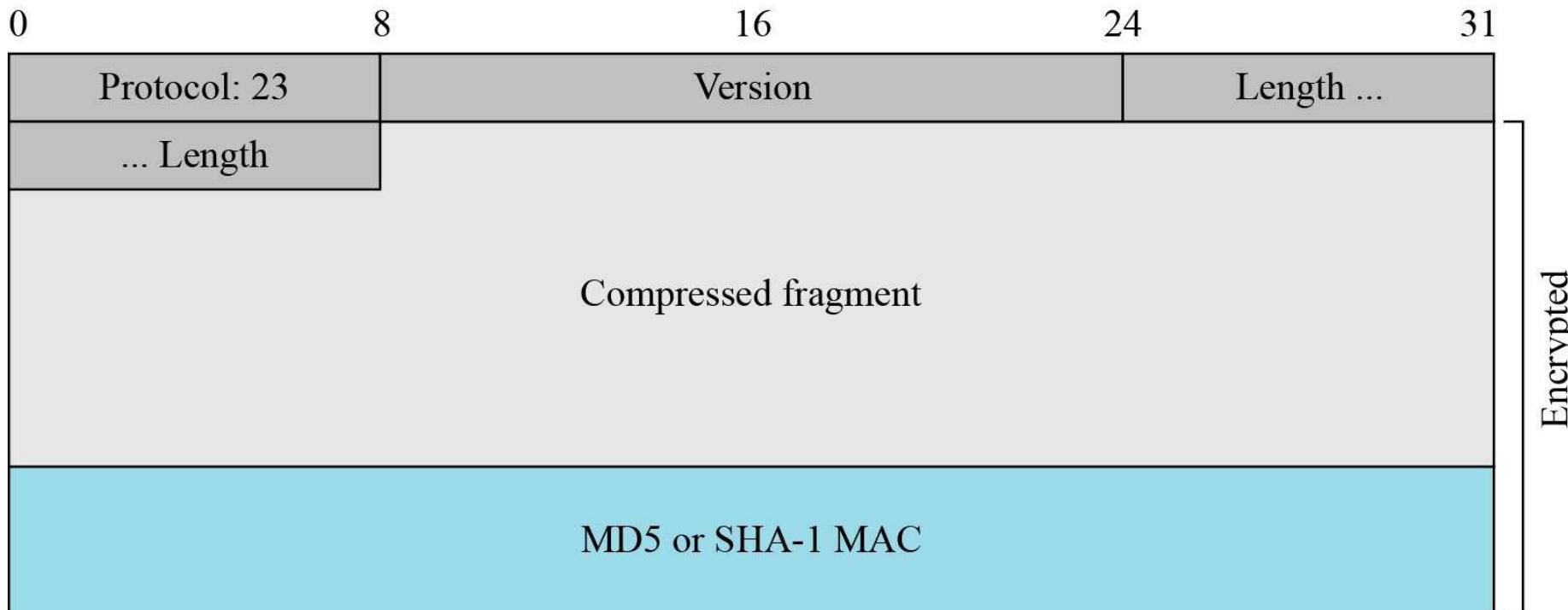
### 17.3.3 Continued

Figure 17.38 Hash calculation for Finished message



### 17.3.3 Application Data

**Figure 17.39 Record Protocol message for application data**



# 17-4 Transport Layer Security (TLS)

*The Transport Layer Security (TLS) protocol is the IETF standard version of the SSL protocol. The two are very similar, with slight differences.*

## Topics discussed in this section:

17.4.1 Version

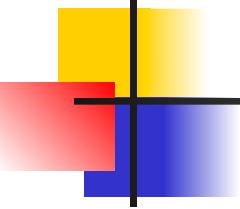
17.4.2 Cipher Suite

17.4.3 Generation of Cryptographic Secrets

17.4.4 Alert Protocol

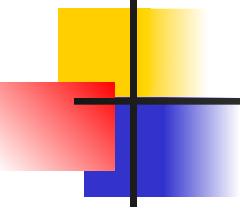
17.4.5 Handshake Protocol

17.4.6 Record Protocol



## *17.4.1 Version*

*The first difference is the version number (major and minor). The current version of SSL is 3.0; the current version of TLS is 1.0. In other words, SSLv3.0 is compatible with TLSv1.0.*



## 17.4.2 Cipher Suite

*Another minor difference between SSL and TLS is the lack of support for the Fortezza method. TLS does not support Fortezza for key exchange or for encryption/decryption. Table 17.6 shows the cipher suite list for TLS (without export entries).*

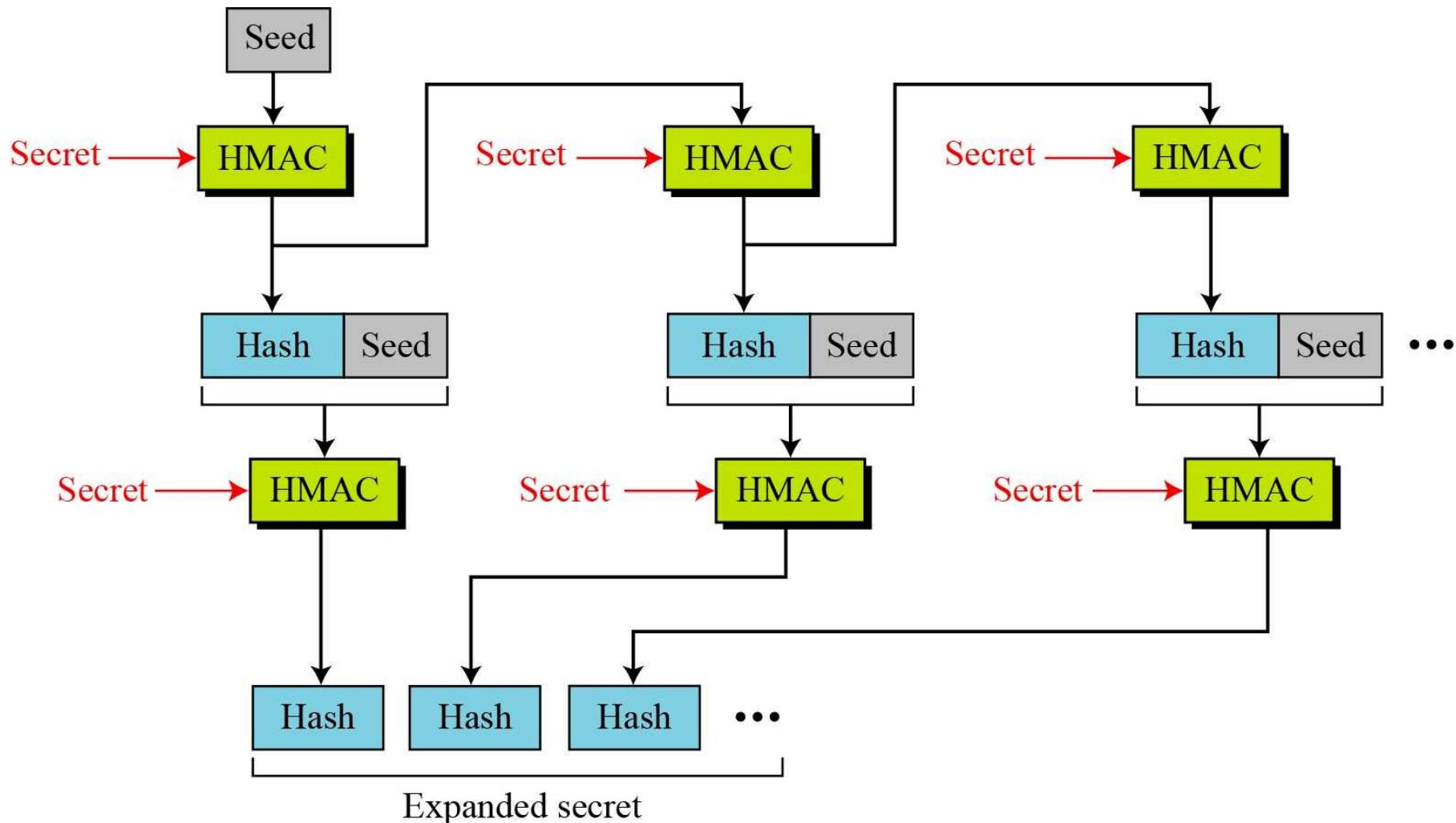
## 17.4.2 Continued

**Table 17.6 Cipher Suite for TLS**

<i>Cipher suite</i>	<i>Key Exchange</i>	<i>Encryption</i>	<i>Hash</i>
TLS_NULL_WITH_NULL_NULL	NULL	NULL	NULL
TLS_RSA_WITH_NULL_MD5	RSA	NULL	MD5
TLS_RSA_WITH_NULL_SHA	RSA	NULL	SHA-1
TLS_RSA_WITH_RC4_128_MD5	RSA	RC4	MD5
TLS_RSA_WITH_RC4_128_SHA	RSA	RC4	SHA-1
TLS_RSA_WITH_IDEA_CBC_SHA	RSA	IDEA	SHA-1
TLS_RSA_WITH DES_CBC_SHA	RSA	DES	SHA-1
TLS_RSA_WITH_3DES_EDE_CBC_SHA	RSA	3DES	SHA-1
TLS_DH_anon_WITH_RC4_128_MD5	DH_anon	RC4	MD5
TLS_DH_anon_WITH DES_CBC_SHA	DH_anon	DES	SHA-1
TLS_DH_anon_WITH_3DES_EDE_CBC_SHA	DH_anon	3DES	SHA-1
TLS_DHE_RSA_WITH DES_CBC_SHA	DHE_RSA	DES	SHA-1
TLS_DHE_RSA_WITH_3DES_EDE_CBC_SHA	DHE_RSA	3DES	SHA-1
TLS_DHE_DSS_WITH DES_CBC_SHA	DHE_DSS	DES	SHA-1
TLS_DHE_DSS_WITH_3DES_EDE_CBC_SHA	DHE_DSS	3DES	SHA-1
TLS_DH_RSA_WITH DES_CBC_SHA	DH_RSA	DES	SHA-1
TLS_DH_RSA_WITH_3DES_EDE_CBC_SHA	DH_RSA	3DES	SHA-1
TLS_DH_DSS_WITH DES_CBC_SHA	DH_DSS	DES	SHA-1
TLS_DH_DSS_WITH_3DES_EDE_CBC_SHA	DH_DSS	3DES	SHA-1

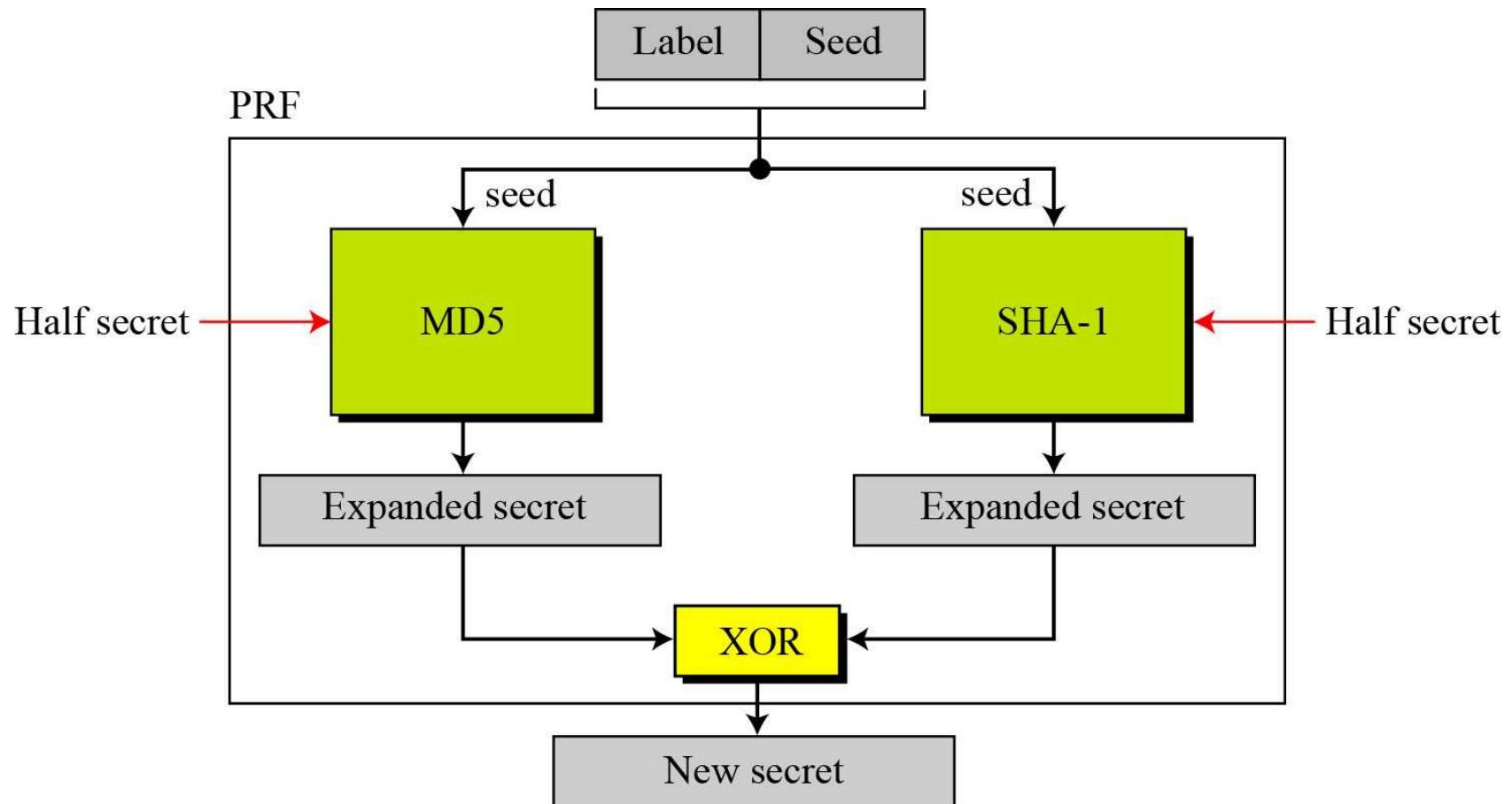
## 17.4.3 Generation of Cryptographic Secrets

Figure 17.40 Data-expansion function



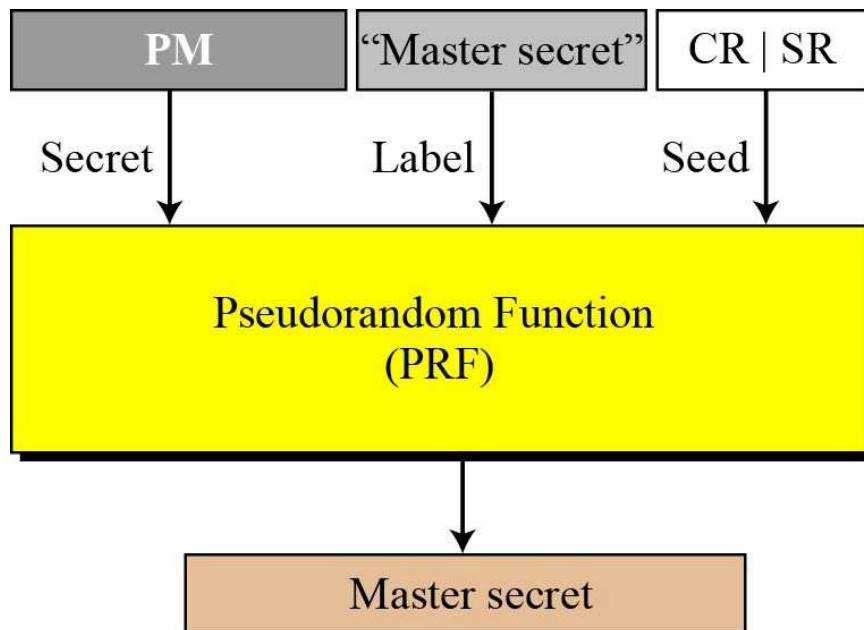
## 17.4.3 Continued

Figure 17.41 PRF



## 17.4.3 Continued

Figure 17.42 Master secret generation



PM: Pre-master Secret

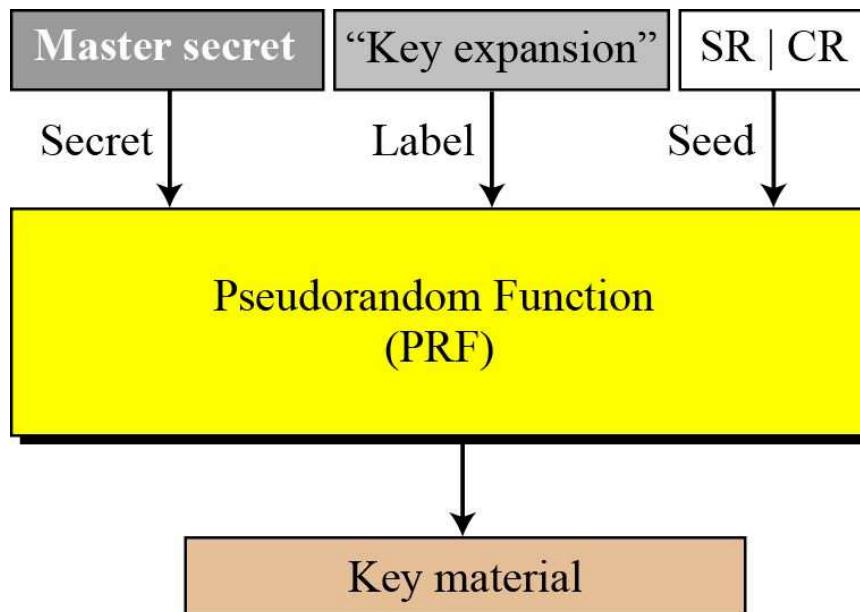
CR: Client Random Number

SR: Server Random Number

|: Concatenation

## 17.4.3 *Continued*

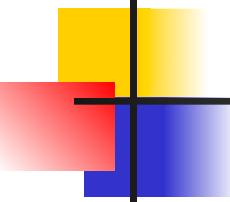
Figure 17.43 *Key material generation*



CR: Client Random Number

SR: Server Random Number

|: Concatenation



## *17.4.4 Alert Protocol*

*TLS supports all of the alerts defined in SSL except for NoCertificate. TLS also adds some new ones to the list. Table 17.7 shows the full list of alerts supported by TLS.*

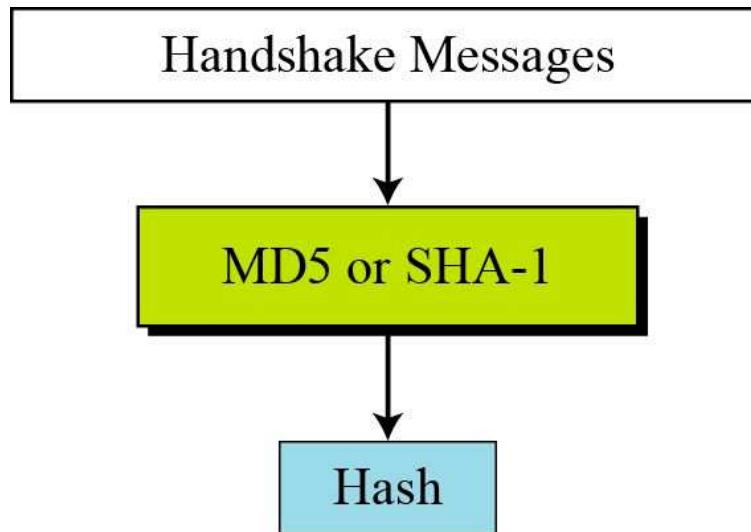
## 17.4.4 Continued

**Table 17.7 Alerts defined for TLS**

<i>Value</i>	<i>Description</i>	<i>Meaning</i>
0	<i>CloseNotify</i>	Sender will not send any more messages.
10	<i>UnexpectedMessage</i>	An inappropriate message received.
20	<i>BadRecordMAC</i>	An incorrect MAC received.
21	<i>DecryptionFailed</i>	Decrypted message is invalid.
22	<i>RecordOverflow</i>	Message size is more than $2^{14} + 2048$ .
30	<i>DecompressionFailure</i>	Unable to decompress appropriately.
40	<i>HandshakeFailure</i>	Sender unable to finalize the handshake.
42	<i>BadCertificate</i>	Received certificate corrupted.
43	<i>UnsupportedCertificate</i>	Type of received certificate is not supported.
44	<i>CertificateRevoked</i>	Signer has revoked the certificate.
45	<i>CertificateExpired</i>	Certificate has expired.
46	<i>CertificateUnknown</i>	Certificate unknown.
47	<i>IllegalParameter</i>	A field out of range or inconsistent with others.
48	<i>UnknownCA</i>	CA could not be identified.

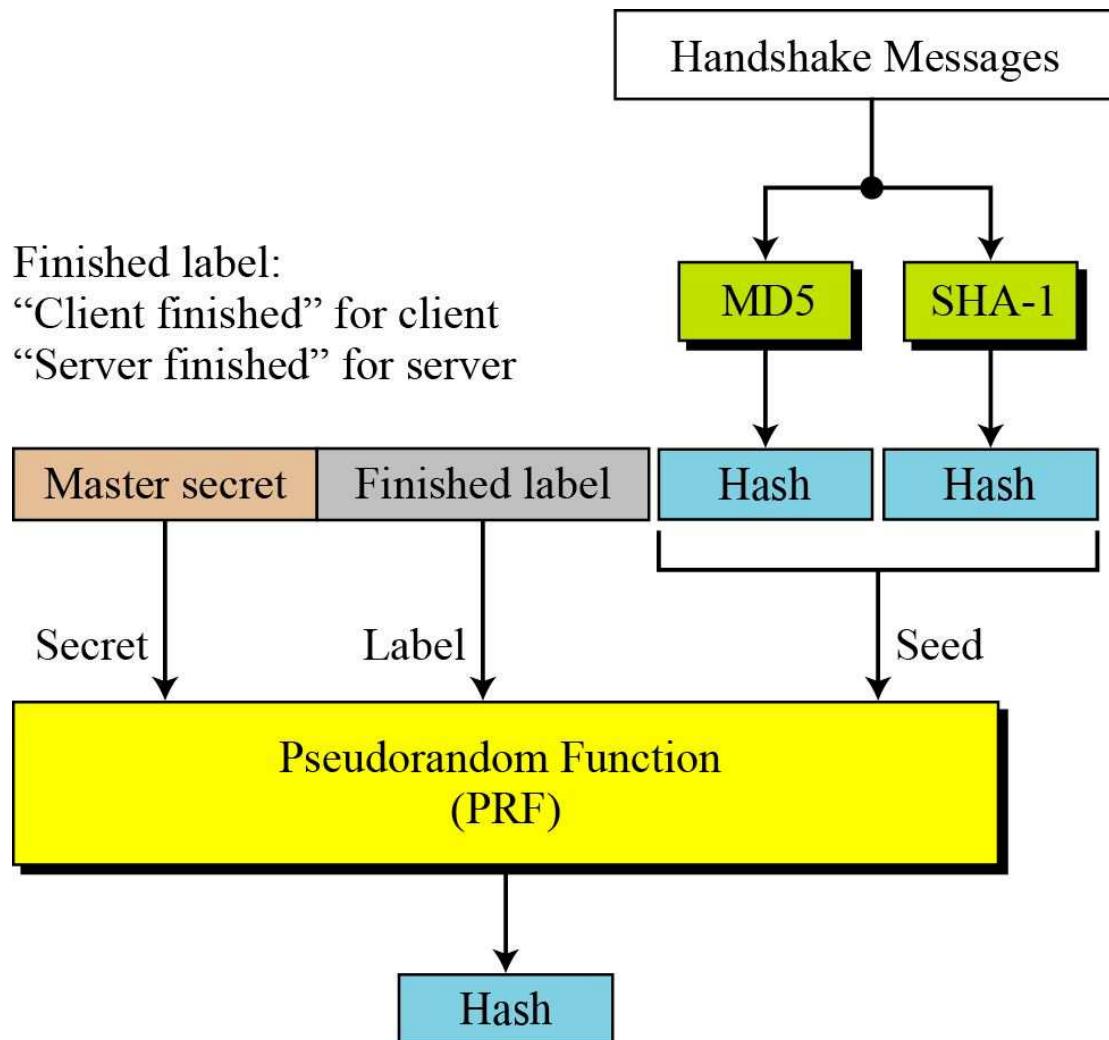
## 17.4.5 Handshake Protocol

**Figure 17.44 Hash for CertificateVerify message in TLS**



## 17.4.5 Continued

Figure 17.45 Hash for Finished message in TLS



## 17.4.6 Record Protocol

Figure 17.46 HMAC for TLS

