

## MODULE 4

Chapters	
1.IEEE 802.11 wireless LAN security	2.viruses,worms and other malware
3.firewalls	4.Intrusion Prevention and detection
5.Web Services Security	

### 1. IEEE 802.11 Wireless LAN Security

- Wireless networks present formidable challenges in the area of security.
- The open nature of such networks makes it relatively easy to sniff packets or even modify and inject malicious packets into the network.
- The ease with which such attacks are launched necessitates careful design and deployment of security protocols for wireless networks.

#### 1.1 BACKGROUND

##### Wired network

- In many organizations, the wired network is an Ethernet LAN with an existing security infrastructure that includes an authentication server (AS).
- **AAA (Authentication/Authorization/Accounting)** functionality is often provided by a RADIUS (Remote Authentication Dial in User Service) server.

##### WLANs(wireless LANs)

- There are *two principal types of WLANs* —
  1. **Ad hoc networks:** where stations (possibly mobile) communicate directly with each other.
  2. **Infrastructure WLANs:** which use an access point (AP) as shown in below figure.

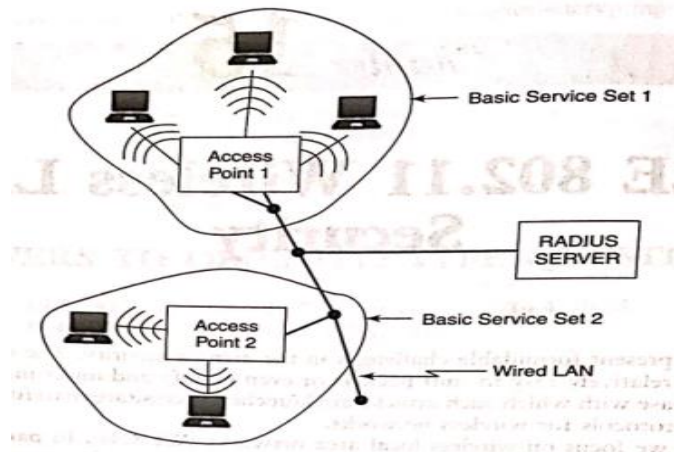


Figure: Infrastructure wireless LAN

### Infrastructure WLANs :

- A station first, sends a frame to an AP and the AP then delivers it to its final destination.
- The destination may be another wireless station or it may be a station on the wired network that the AP is connected to.
- The **AP** thus serves as a **bridge** between the WLAN and the existing wired network.
- The challenge then is to develop protocols that seamlessly integrate the WLAN with the security infrastructure of the wired network.
- A network of wireless stations associated with an AP is referred to as a **basic service set**. Such a network may be adequate for a home or small enterprise.
- The union of the basic service sets comprises an **extended service set (ESS)**.
- Each station and AP in the ESS is uniquely identified by a MAC address — **a 48-bit quantity**.
- Each AP is also identified by an **SSID (service set ID)**, which is a character string of length at most **32 characters**.
- A wireless station, on power-up, needs to first discover an AP within its range.

- |   |   |
|---|---|
| <ul style="list-style-type: none"> <li>➤ This can be done by monitoring the wireless medium for a special kind of frame called a <b>Beacon</b>, which is periodically <b>broadcast by the AP</b>.</li> <li>➤ The Beacon usually contains the <b>SSID</b> of the broadcasting AP.</li> </ul> | <ul style="list-style-type: none"> <li>➤ Alternatively, a station may send a <b>Probe Request frame</b>, which probes for APs within its range.</li> <li>➤ An AP, on hearing such a request, responds with a Probe Response frame.</li> <li>➤ Like the Beacon, the <b>Probe Response frame</b> contains the SSID of the AP and also information about its capabilities, supported data rates, etc.</li> </ul> |
|---|---|



- A station that wishes to associate with an AP sends it an **Associate Request frame**.

- The AP replies with an **Associate Response frame** if it accepts the request for associating with it.
  1. The earliest protocol that incorporated security in WiFi was **WEP**.
    - ✓ Designed to provide authentication/access control, data integrity, and confidentiality, it failed on all three counts.
  2. **WiFi Protected Access (WPA)**
    - ✓ WPA was intended to fix the shortcomings of WEP without requiring new wireless network cards.
    - ✓ But WPA is not perfect — it too is susceptible to attacks on its cryptographic algorithms.
  3. **WPA2**
    - ✓ All the deficiencies in WEP have been addressed in the IEEE 802.11i (implemented in WPA2) .

## 1.2 AUTHENTICATION

### 1.2.1 Pre-WEP Authentication

- **Drawbacks:**
  1. An attacker could easily sniff the value of SSID from frames such as the beacon or probe response and then use it for authentication.
  2. Another approach was to restrict admission to the WLAN by MAC address.
    - ✓ The AP would maintain a list of MAC addresses (access control list) of stations permitted to join the WLAN.
    - ✓ valid MAC addresses could be obtained by sniffing the wireless medium.
    - ✓ The attacker could then modify his network card to spoof a valid MAC address. So, neither of these approaches was truly secure.

### 1.2.2 Authentication in WEP

- In WEP, the station authenticates itself to the AP using a challenge—response protocol .
- Basically, the AP generates a challenge (nonce) and sends it to the station.
- The station encrypts the challenge and sends it to the AP.
- The stream cipher, RC4, is used for encryption.
- **Response From Station:** the station computes a keystream, which is a function of a 40-bit shared secret, S, and a 24-bit Initialization Vector (IV).
- The challenge is then XORed with the keystream to create the response.

### **RESPONSE = CHALLENGE (XOR) KEYSTREAM(S, IV)**

- The response together with the IV is sent by the station to the AP.
- The shared secret, S, is common to all stations authorized to use the WLAN.

#### **Drawbacks:**

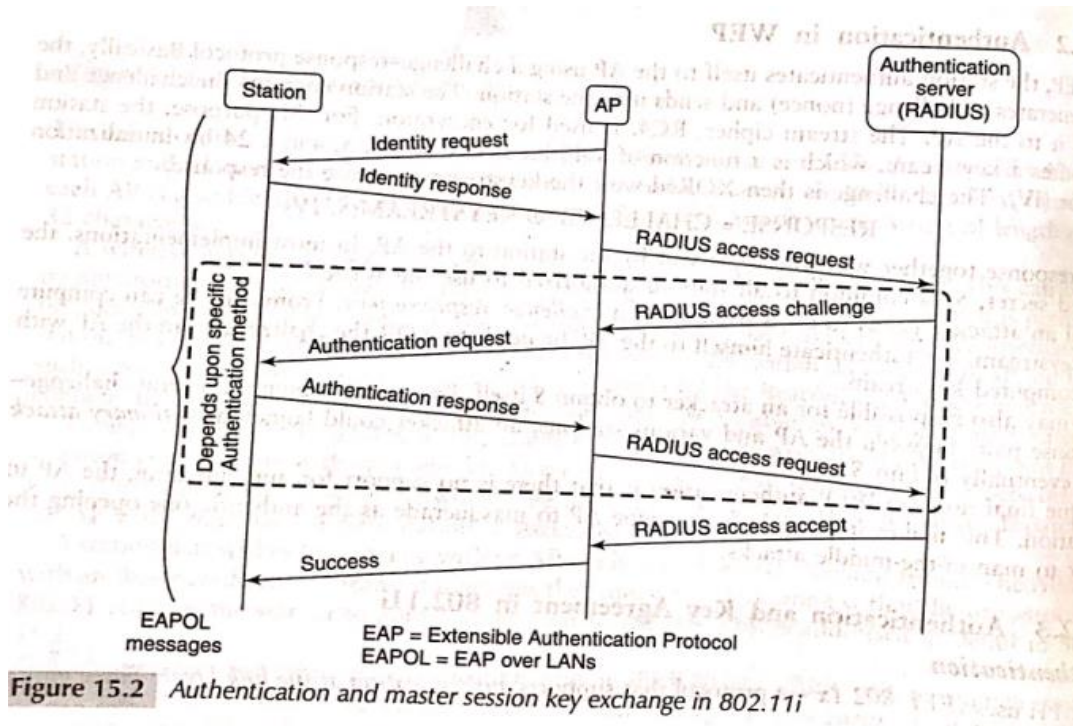
- All an attacker needs to do is to monitor a challenge—response pair.
- From this, he can compute the keystream.
- To authenticate himself to the AP, he needs to XOR the challenge from the AP with the computed keystream.
- It may also be possible for an attacker to obtain S itself.
- By eavesdropping on several challenge—response pairs between the AP and various stations, an attacker could launch a **dictionary attack** and eventually obtain S.

### 1.2.3 Authentication and key agreement in 802.11

- 802.11i uses IEEE 802.1x — a protocol that supports authentication at the link layer.
- Three entities are involved:
  1. **Supplicant (the wireless station)**
  2. **Authenticator (the AP in our case)**
  3. **Authentication server**
- Different authentication mechanisms and message types are defined by the ***Extensible authentication Protocol (EAP)*** standardized by Internet Engineering Task Force (IETF).
- EAP is not really an authentication protocol but rather a ***framework*** upon which various authentication protocols can be supported.
- EAP exchanges are mostly comprised of **requests and responses**.
- For example one party requests the ID of another party.
- The latter responds with its user\_name or e-mail address.
- EAP also defines messages that may contain challenges and responses used in authentication protocols.
- The AP broadcasts its security capabilities in the Beacon or Probe Response frames.
- The station uses the Associate Request frame to communicate its security capabilities.
- 802.11i authentication takes place after the station associates with an AP.

### IEEE 802.11i

- The generic authentication messages in IEEE 802.11i are shown in Fig. 15.2.
- The protocol used between the **station and the AP is EAP** but that used between the AP and the authentication server depends upon the specifics.
- For example, the authentication server is often a RADIUS server which uses its own message types and formats. (RADIUS stands for Remote Authentication Dial in User Service. It is a client—server protocol used for authentication, authorization, and accounting.)



**Figure 15.2 Authentication and master session key exchange in 802.11i**

**EAP = Extensible Authentication Protocol messages**

**EAPOL = EAP over LANs**

➤ The main authentication methods supported by EAP include the following:

1. **EAP-MDS**
2. **EAP-TLS**
3. **EAP-TTLS**
4. **EAP-PEAP**

#### 1. **EAP-MDS**

- ✓ This is most basic of the EAP authentication methods.
- ✓ Here, the authentication server challenges the station to transmit the MD5hash of the user's password.
- ✓ The station prompts the user to type his/her password.
- ✓ It then computes the hash of the password and sends this across.
- ✓ This method is insecure since an attacker could eavesdrop on such a message exchange and then replay the hashed password thus impersonating the owner of the password.
- ✓ Also, this method does not support authentication of the AP to the station.

## 2. EAP-TLS

- ✓ EAP-TLS is based on the SSL/TLS protocol
- ✓ most secure and provides mutual authentication and agreement on a **master session key**.
- ✓ It requires the AP as well as the user (station) to have digital certificates.
- ✓ It is relatively straightforward to equip each AP with a digital certificate and a corresponding private key but extending the Via to each user of the WLAN may not be feasible.

## 3. EAP-TTLS

- ✓ (tunnelled TLS) requires certificates only at the AP end.
- ✓ The AP authenticates itself to the station and both sides construct a secure tunnel between themselves.
- ✓ Over this secure tunnel, the station authenticates itself to the AP.
- ✓ The station could transmit **attribute-value** pairs such as  
**user\_name = akshay**  
**password = 4rP#mNaS&7**

## 4 Protected EAP (PEAP)

- ✓ This was proposed by Microsoft, Cisco, and RSA Security, is very similar to EAP-TTLS.
- ✓ In PEAP, the secure tunnel is used to start a second EAP exchange where in the station authenticates itself to the authentication server.
- ✓ The enhanced security offered by EAP-TLS, EAP-TTLS, and PEAP does, however, come at a steep price in performance measured by the message and computational overheads incurred during authentication.

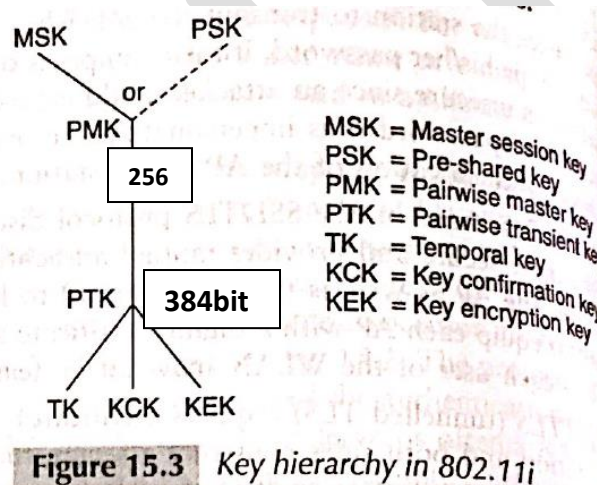
### **Key Hierarchy**

- There are two types of keys used in WLANs.
- The first are ***pairwise keys*** used to protect traffic between a station and an AP.
- The second type of key is the ***group key*** intended to protect broadcast or multicast traffic between an AP and multiple stations.

The hierarchy of 802.11i keys:

- The root of the key hierarchy is the ***Pairwise Master Key (PMK)***.
  - ✓ This is obtained in one of two ways
  - ✓ The station and the authentication server may agree on a Master Session Key (MSK) as part a of the authentication procedure.
  - ✓ The authentication server communicates this key to AP
  - ✓ The AP and station then derive the PMK from the MSK.

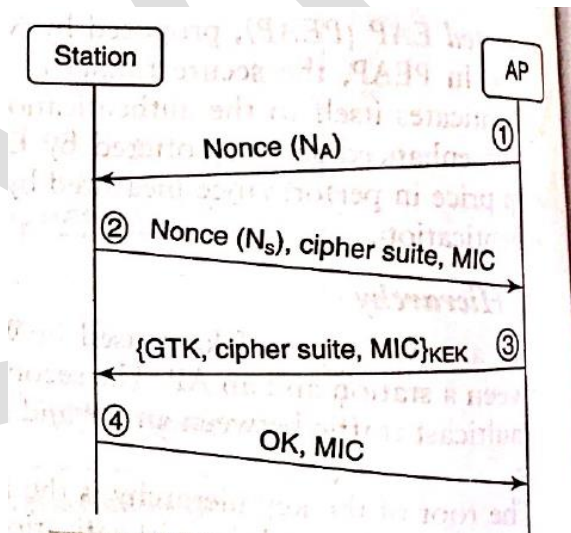
- An alternative to computing a fresh PMK for each session is the **Pre-Shared Key, (PSK)**, which is used as the PMK.
- **Pairwise Transient Key (PTK).**
  - ✓ The 256-bit PMK is used to *derive* a **384-bit pairwise Transient Key (PTK)**.
  - ✓ The PTK is a pseudo random function of the PMK.
  - ✓ PRF(nonce of AP,nonce of station,MAC address of AP,MAC address of station, PMK).
- Three **128 bit** chunks are extracted from the 384 bit PTK for the following purposes:
  1. A **Temporal Key (TK)** is used for both encryption and integrity protection of data between the AP and the station.
  2. A **Key Confirmation Key (KCK): Integrity protection is supported by a MAC computed as a function of the message and** and Associate Request Frames and**the KCK.**
  3. A **Key Encryption Key (KEK)** is used to encrypt the message containing the group key.



## Four-way Handshake

- The main goals of the four-way handshake are to
  - (a) Derive the PTK from the PMK,
  - (b) Verify the cipher suites communicated in the Beacon and Associate Request Frames.
  - (c) Communicate the group keys from the AP to the station.
- Figure 15.4 shows the messages comprising the four-way handshake.
  1. Message 1: The AP first sends a nonce,  $N_A$ , to the station.
  2. Message 2:
    - ✓ The station chooses a nonce,  $N_s$  station computes the PTK as follows
    - ✓  **$PTK = \text{prf}(PMK, N_A, N_s, MAC_A, MAC_S) \dots$**
    - ✓ The PTK is a pseudo-random function (prf) of the PMK, the MAC addresses of the station and AP and nonces contributed by the station and the AP.

- ✓ The two nonces help prevent replay attacks.
  - ✓ Three 128-bit keys — **TK, KCK, and KEK** are extracted from the 384-bit PTK (Fig. 15.3).
  - ✓ The station sends nonce ,cipher suite and uses KCK to compute MIC(message integrity check).
3. Message 3 :
- ✓ On receiving the message 2,AP computes the PTK from the above expression used by the station .
  - ✓ AP verifies the integrity and source of message 2 using the key KCK.
  - ✓ Message 3 contains group transient key (GTK).this is the key used by the AP ans all staions to integrity protect and multicast and broadcast.
4. Message 4:
- ✓ This is an acknowledgement from the station that it has received the previous messages without error.
  - ✓ It is a signal to the AP that henceforth all messages will be integrity-protected and encrypted with the TK.



**Figure 15.4** Four-way handshake in 802.11i



## 1.3 CONFIDENTIALITY AND INTEGRITY

### 1.3.1 Data Protection in WEP

- WEP was designed to provide message confidentiality, integrity, and access control but it failed on all three counts.
- In this section, we show how plaintext can be recovered and messages can be modified due to flawed design decisions in WEP.
- There are many lessons to be learned from WEP — the most important being how not to design protocols for security.

### WEP Encryption and Integrity Checking

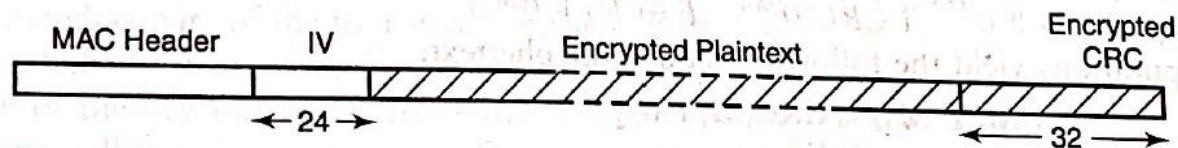
- WEP uses the stream cipher, RC4, for encrypting messages.
- It generates a pseudo-random keystream, KS, which is a function of a static secret shared between the two communicating parties.
- In order to have KS vary from message to message, a random per-message initialization vector, IV, is also used to generate KS.
- Early implementations of WEP used a 40-bit secret, S, concatenated with a 24-bit IV to create, in effect, a "64-bit key."
- KS is xored with the plaintext, P, to obtain the ciphertext, C or

$$C = P \oplus KS(S, IV)$$

The plaintext p is:

$$P = C \oplus KS(S, IV)$$

- 



5 WEP frame

### Known plaintext attack

- The first problem with WEP is the possibility of keystream re-use.
- Since the IV is 24 bits in length, there are only  $2^{24}$  distinct keystreams that could be constructed given a secret S.
- Suppose an attacker finds two frames which were encrypted using the same IV.
- Let their ciphertexts be C and C'.
- Let the corresponding plaintexts be P and P'.

$$P \oplus P' = C \oplus C'$$

$$P' = P \oplus C \oplus C'$$

- Thus knowing  $c, c'$ , and  $p$ , we can obtain  $p'$  which is called as known plaintext attack.

### Message modification

- Consider an attacker who wishes to modify a message sent by a legitimate user.
- Let the sender's plaintext (not including the CRC checksum) be  $M_1 F M_2$  where  $M_1, F$ , and  $M_2$  are each binary strings.
- The attacker wishes to substitute the substring,  $F$ , with another substring,  $F'$ ,
- so that the decrypted message seen by the receiver is  $M_1 F' M_2$ . The attacker does not need to know the values,  $M_1$  and  $M_2$ . However, we assume that he knows  $F$  and  $F'$ .
- Ideally, the message integrity check should detect any modification to an existing message. Can the attacker modify the message (including checksum) in such a way so that the modification is undetected at the receiver end?
- For the above plaintext, the **ciphertext** computed by the sender is :

$$CT = ((M_1 F M_2) \parallel CRC(M_1 F M_2)) \oplus KS$$

The attacker intercepts the ciphertext and performs the following operations:

1. He first constructs the string,  $0^{|M_1|} \parallel (F \oplus F') \parallel 0^{|M_2|}$ . Here,  $0^{|M_1|}$  is a string of  $|M_1|$  zeros where  $|x|$  is the length of the substring  $x$ .
2. He then computes the CRC on this string,  $CRC(0^{|M_1|} \parallel (F \oplus F') \parallel 0^{|M_2|})$ .
3. He finally XORs the original ciphertext with  $0^{|M_1|} \parallel (F \oplus F') \parallel 0^{|M_2|} \parallel CRC(0^{|M_1|} \parallel (F \oplus F') \parallel 0^{|M_2|})$ .

The last step follows from the fact that the CRC is a linear operation, i.e.,

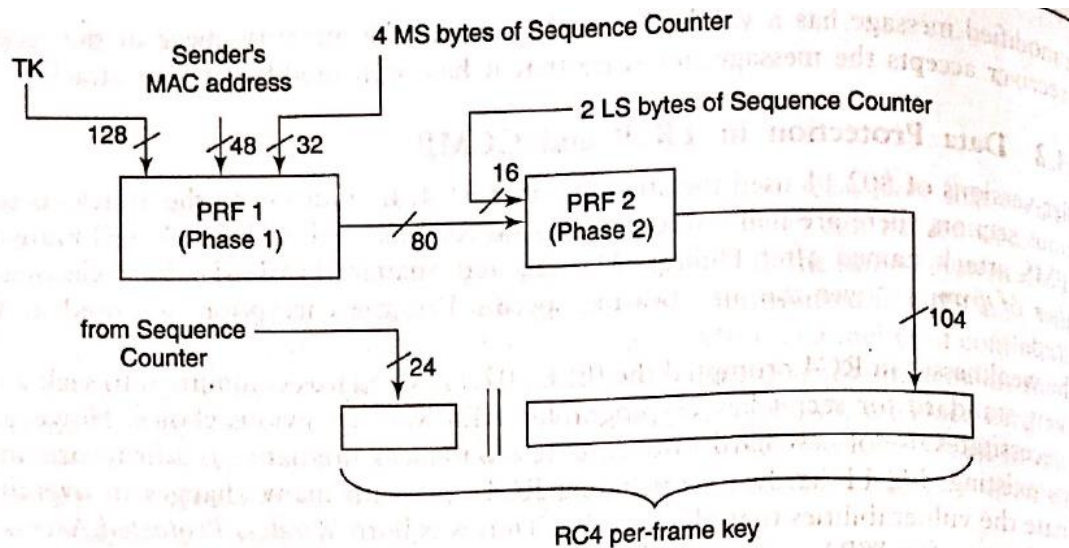
$$CRC(m_1 \oplus m_2) = CRC(m_1) \oplus CRC(m_2)$$

The receiver, on decrypting the ciphertext, obtains

$$(M_1 F' M_2) \parallel CRC(M_1 F' M_2)$$

- The modified message has a **valid CRC** and so passes the integrity check at the receiver.
- Hence, the receiver accepts the message, **unaware that it has been modified by an attacker.**

### 1.3.2 Data protection in TKIP and CCMP



**Figure 15.6** Two-phase key mixing in TKIP

- The technical name for WPA is *Temporal Key Integrity Protocol* (TKIP).
- By contrast, the encryption key in TKIP is 128 bits.
- TKIP generates a random and different encryption key for each frame sent. It employs a process called *two-phase key mixing*.
- The inputs to this process are the 128-bit temporal key, TK, computed as part of the four-way handshake, the sender's MAC address and the four most significant bytes of a 48-bit *frame sequence counter*.
- The randomizing capability of the key mixing function and the large size of the key space virtually guarantee that "keystream collisions" never occur.
- Thus, known plaintext attacks that could be successfully launched on WEP have no chance of success with TKIP.
- The sequence counter is incremented for each frame sent.
- It is also carried in the header of each frame.
- This helps protect the receiver from *replay attacks*.
- Figure 15.6 shows the two phases used in generating the **RC4 key**.
- Two pseudo-random function (PRF1 and PRF2) are employed in the two phases.
- The 32 most Significant bits of the sequence counter are input to PRF1.
- The least significant 16 bits of the sequence counter are inputs to PRF2. So, the output of PRF2 changes for each frame sent.
- MIC is computed as a function of the data in the frame and also some fields in the MAC header such as the source and destination addresses.

- It also uses as input a key derived from the PTK which was computed during the four-way handshake.
- Due to design constraints on WEP cards, MIC's implementation uses simple logical functions, shifts, etc. Hence, it is not as secure as a keyed cryptographic hash.
- On the other hand, it is much better compared to the CRC checksum used in WEP.

## CCMP

- The implementation of 802.11i that uses AES is referred to as WPA-2.its technical name is counter mode with CBC MAC protocol(CCMP).
- In CCMP terminology, this count is referred to as a packet number (PN).
- The count is maintained at both sender and receiver ends.
- The PN is also included in a special CCMP header field in a CCMP frame.
- The PN is incremented by the sender after each frame is sent.
- Upon receipt of a fresh frame in that session, the receiver compares the value of PN in the CCM header versus the value stored by it.
- If the value is less than the stored value, the frame is likely to be a replayed frame and is hence discarded.
- The first task in preparing a frame for transmission is to compute a MIC.
- The **MIC** is computed using AES in **Cipher Block Chaining (CBC) mode** with block size 128 bits.
- The key for performing encryption in each stage of Fig below is TK(temporal key).
- The IV for the MIC computation is a "nonce," which includes the 48-bit PN.
- The second and third blocks used in the MIC computation are specific fields in the frame header such as the MAC addresses, sequence control, and frame type.
- Next, the blocks in the frame data are sequentially processed resulting in an 8-byte MIC.
- The next step is encryption.
- The frame data and the MIC are concatenated and then encrypted using AES in counter mode (Fig. 15.7).
- Let  $n$  be the total number of blocks in the **frame body + MIC**.
- The procedure for encrypting the  $i$ -th block is:
  - Compute  $A_i = E_{TK}(PN + i*j)$ . Here, PN is the packet number and  $j$  is a constant known to both sender and receiver.
  - Compute  $i$ -th block of ciphertext =  $A \text{ (xor) } P_i$ .
  - Here,  $P_i$  is the  $i$ -th block of plaintext.
- The frame now includes two new fields — the CCMP header and the MIC.
- Upon receipt of the frame, the receiver reverses the operations performed by the sender.
- It performs decryption followed by MIC verification.

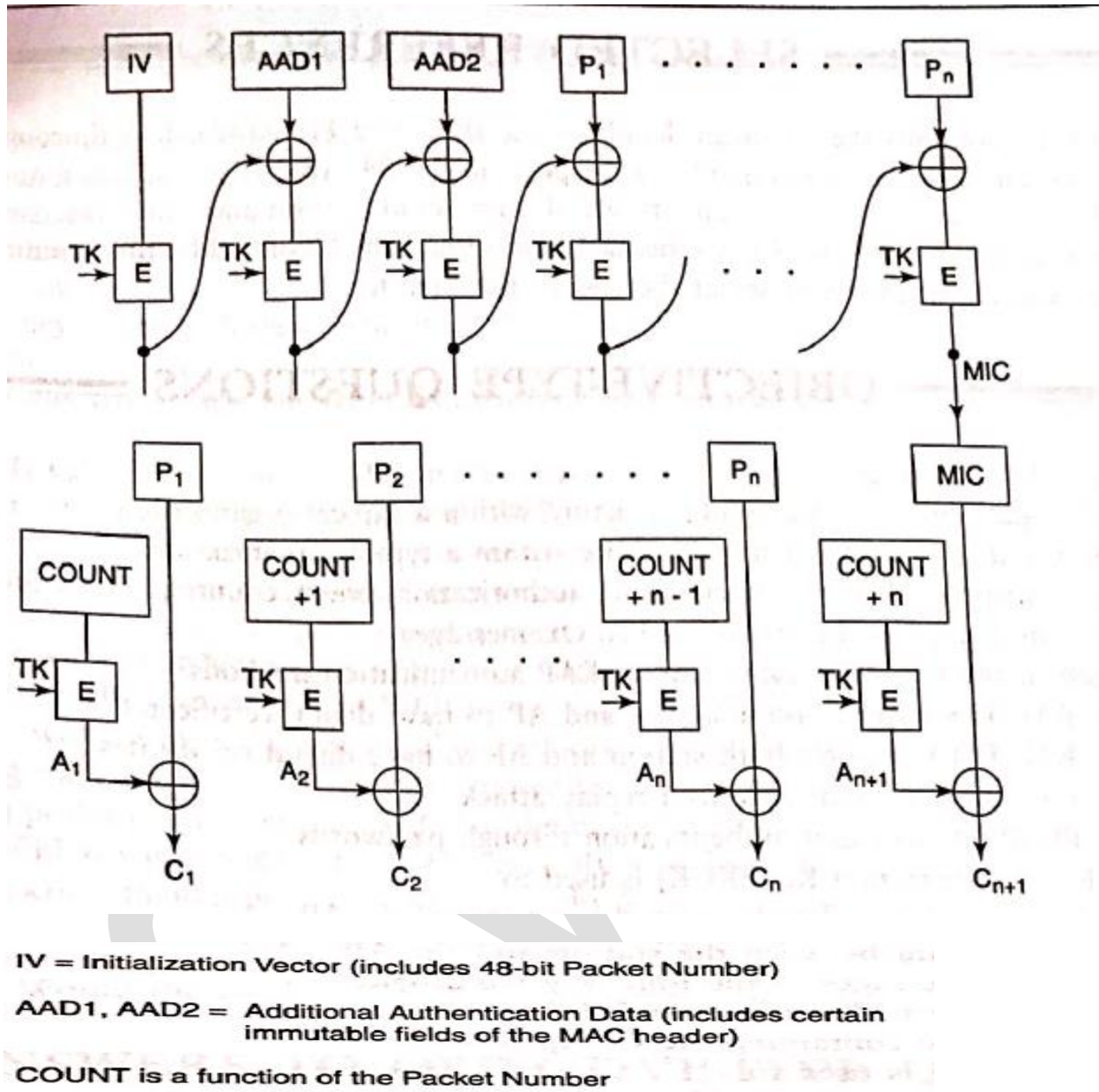


Fig : MAC generation and encryption in CCMP

## Firewalls

- **Definition:** A firewall acts as a *security* guard controlling access between an internal protected network and an external untrusted network based on a given security policy.
- Besides preventing intruders getting in, a firewall also helps prevent confidential inside data from getting out.
- A firewall may be implemented in hardware as a *stand-alone "firewall appliance" or in software on a PC.*
- A single firewall may be adequate for small businesses and homes. However, in several large enterprises, multiple firewalls are deployed to achieve *defence in depth.*

### 2.1 BASICS

#### 2.1.1 Firewall Functionality

- The main functions of a firewall are listed as follows:
- **Access Control:**
  - ✓ A firewall filters incoming (from the Internet into the organization) as well as outgoing (from within the organization to the outside) packets.
  - ✓ A firewall is said to be configured with a rule set based on which it decides which packets are to be allowed and which are to be dropped.
- **Address/Port Translation.**
  - ✓ NAT was initially devised to alleviate the serious shortage of IP addresses by providing a set of private addresses that could be used by system administrators on their internal networks but that are globally invalid (on the Internet).
  - ✓ it is possible to conceal the addressing schema of these machines from the outside world through the use of NAT.
  - ✓ Through NAT, internal machines, though not visible on the Internet, can establish a connection with external machines on the Internet. NATing is often done by firewalls.
- **Logging.**
  - ✓ A sound security architecture will ensure that each incoming or outgoing packet encounters at least one firewall.
  - ✓ The firewall can log all anomalous packets or flows for later study.
  - ✓ These logs are very useful for studying attempts at intrusion together with various worm and DDoS attacks.
- **Authentication, Caching,** etc. Some types of firewalls perform authentication of external machines attempting to establish a connection with an internal machine.
- A special type of firewall called *web proxy* authenticates internal users attempting to access an external service. Such a firewall is also used to cache frequently requested webpages. This results in decreased response time to the client while saving communication bandwidth.

### 2.1.2 Policies and Access Control Lists

- High-level policies for access to various types of services are formulated within an organization or campus. Examples of these include the following:
  - ✓ All received e-mail should be filtered for spam and viruses.
  - ✓ All HTTP requests by external clients for access to authorized pages of the organization's website should be permitted.
  - ✓ DNS queries made by external clients should be allowed provided they pertain to addresses of the organization's publicly accessible services such as the web server or the external e-mail server. However, queries related to the IP addresses of internal machines should not be entertained.
  - ✓ The organization's employees should be allowed to remotely log into authorized internal machines. However, all such communication should be authenticated and encrypted.
  - ✓ Only two types of outgoing traffic are permitted. First, all e-mail from within the organization to the outside world are permitted. Second, requests emanating from within the organization for external webpages are permitted. However, requests for pages from certain "inappropriate" websites should be denied.
- **High-level policies are translated into a set of rules that comprise an Access Control List.**
- A rule specifies the action to be taken as a function of
  - (i) **the packet's source IP address and port number**
  - (ii) **the packet's destination IP address and port number**
  - (iii) **the transport protocol in use (TCP or UDP)**
  - (iv) **the packet's direction — incoming or outgoing**
- The Access Control List for the high-level policies is described in Table 21.1.
- Policies can, in general, be either **permissive or restrictive**.
- **A permissive policy is defined as follows:**
  - ✓ **Permit all packets except those that are explicitly forbidden.**
- **A restrictive policy, on the other hand, is defined as follows:**
  - \* **Drop all packets except those that are explicitly permitted.**
- The ACL in Table 21.1 implements a restrictive policy — the default action is Deny as expressed in rules 5 and 8.
- The rules are scanned top to bottom.
- As soon as a rule is found that matches the packet's attributes (IP addresses, port numbers, etc.), the action in that rule (usually permit or deny) is taken and no further rules are processed for that packet.
- The scanning order is important.
- For example, if rules 4 and 5 in Table 21.1 are interchanged, then IPSec traffic will be dropped.
- Also, from a performance perspective, it makes sense to put the most frequently acted upon rule earlier on.
- By so doing, we can expedite the decision on what to do with a packet.

- Finally, it is important to include the default deny rule at the end of the rule set — this prevents ambiguity over what action to take for a packet that has not been matched against the attributes in any of the previous rules.

**Table 21.1** Example access control list

No.	In-bound (I) or out-bound (O)	Transport protocol	Src. IP addr.	Src. port	Dest. IP addr.	Dest port	Action	Comment
1	I	TCP	Any	Any	MS	25	Permit	Allow incoming e-mail
2	I	TCP	Any	Any	WS	80	Permit	Allow requests for organization's webpages
3	I	UDP	Any	Any	NS	53	Permit	Allow DNS queries
4	I	IPSec	Any	Any	*	*	Permit	Allow incoming VPN traffic
5	I	Any	Any	Any	Any	Any	Deny	Forbid all other incoming traffic
6	O	TCP	Any	Any	Any	25	Permit	Allow outgoing e-mail
7	O	TCP	Any	Any	*	80	Permit	Allow requests for external webpages
8	O	Any	Any	Any	Any	Any	Deny	Forbid all other outgoing traffic

**Note:** MS, WS, and NS are the IP addresses of the organization's e-Mail Server, Web Server, and DNS server (Name Server), respectively. \* depends on configuration

### 2.1.3 Firewall Types

- Firewalls can be classified into the categories

1. Packet Filters
2. Stateful Inspection
3. Application Level Firewalls

#### 1. Packet Filters

- This involves checking for matches in the IP, TCP, or UDP headers.
- For example, it may be necessary to check whether a packet carries a certain specific source or destination IP address or port number.
- It is often performed by the border router or access router that connects the organization's network to the Internet.
- In effect, the border router becomes the first line of defence against malicious incoming packets.
- why the packet filtering firewall is inadequate????

Drawbacks:



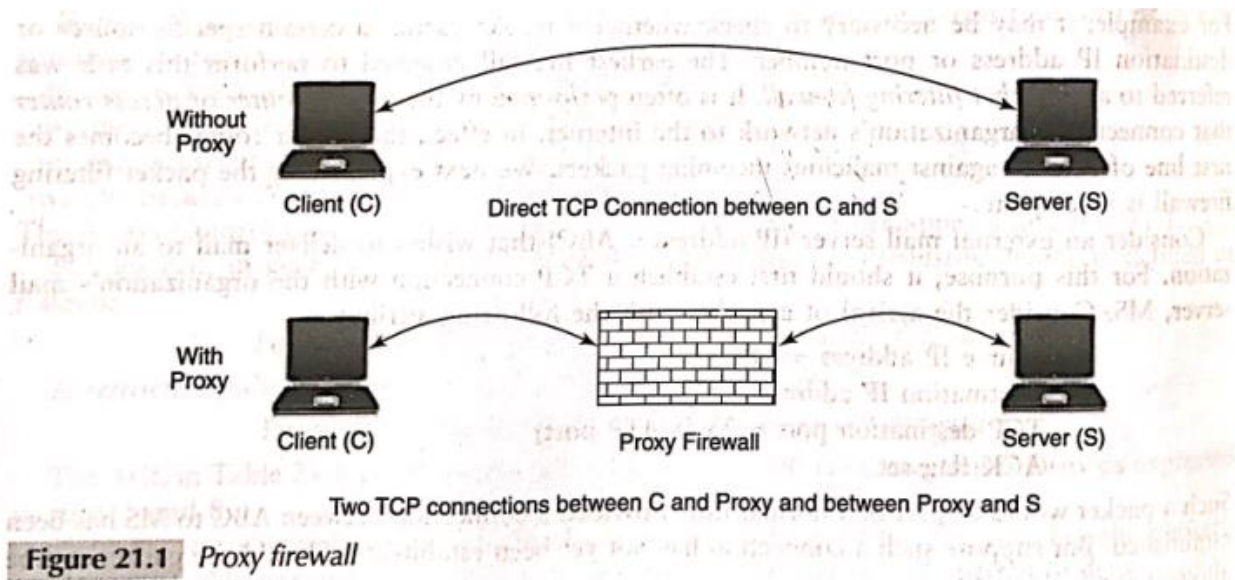
- Consider an external mail server (IP address = ABC) that wishes to deliver mail to an organization.
- For this purpose, it should first establish a TCP connection with the organization's mail server, MS.
- Consider the arrival of a packet with the following attributes:
  - Source IP address = ABC
  - Destination IP address = MS
  - TCP destination port = 25 (SMTP port)
  - ACK flag set
- Such a packet would be part of a normal flow provided a connection between ABC to MS has been established. But suppose such a connection has not yet been established.
- Should the packet still be allowed in? The simple packet filter will allow the packet to enter even if no prior connection between ABC and MS was established.
- It should be noted that such packets are often used to perform port scans.
- A simple packet filter merely inspects the headers of an incoming packet in isolation. It does not view a packet as part of a connection or flow. Hence, it will not be able to filter out such packets arriving from ABC.

## 2. **Stateful Inspection**

- A firewall uses packet's TCP flags and sequence/acknowledgement numbers to determine whether it is part of an existing, authorized flow.
- If it is participating in the establishment of an authorized connection or if it is already part of an existing connection, the packet is permitted, otherwise it is dropped.
- In the above example of the packet from ABC, the stateful packet inspection firewall will realize that it has not encountered the first two packets in the three-way handshake and will hence drop this packet.

## 3. **Application Level Firewalls**

- A packet-filtering firewall, even with the added functionality of stateful packet inspection, is still severely limited.
- What is needed is a firewall that can examine the application payload and scans packets for worms, viruses, spam mail, and inappropriate content. Such a device is called a **deep inspection firewall**.
- A special kind of application-level firewall is built using proxy agents. Such a "***proxy firewall***" acts as an ***intermediary*** between the client and server.
- The client establishes a TCP connection to the proxy and the proxy establishes another TCP connection with the server as shown in Fig. 21.1.
- To a client, the proxy appears as the server and to the server, the proxy appears as the client. Since there is no direct connection between the client and the server, worms and other malware will not be able to pass between the two, assuming that the proxy can detect and filter out the malware. Hence, the presence of the proxy enhances security.



Two TCP connections between C and Proxy and between Proxy and S  
Figure 21.1 Proxy firewall

- There are proxy agents for many application layer protocols including HTTP, SMTP, and FTP.
- In addition to filtering based on application layer data, proxies can perform client authentication and logging.
- An HTTP proxy can also cache webpages.
- Caching has a major impact on performance.
- If the webpage is cached in a web proxy server located in the client's organization, the response time could be greatly reduced compared to that where the page has to be fetched from the external web server.
- Also, caching reduces the demand on external communication bandwidth while easing the load on the web server.
- Firewalls are a necessary element in the security architecture of an organization that permit access to/from the external world. In the next section, we study firewall deployment.

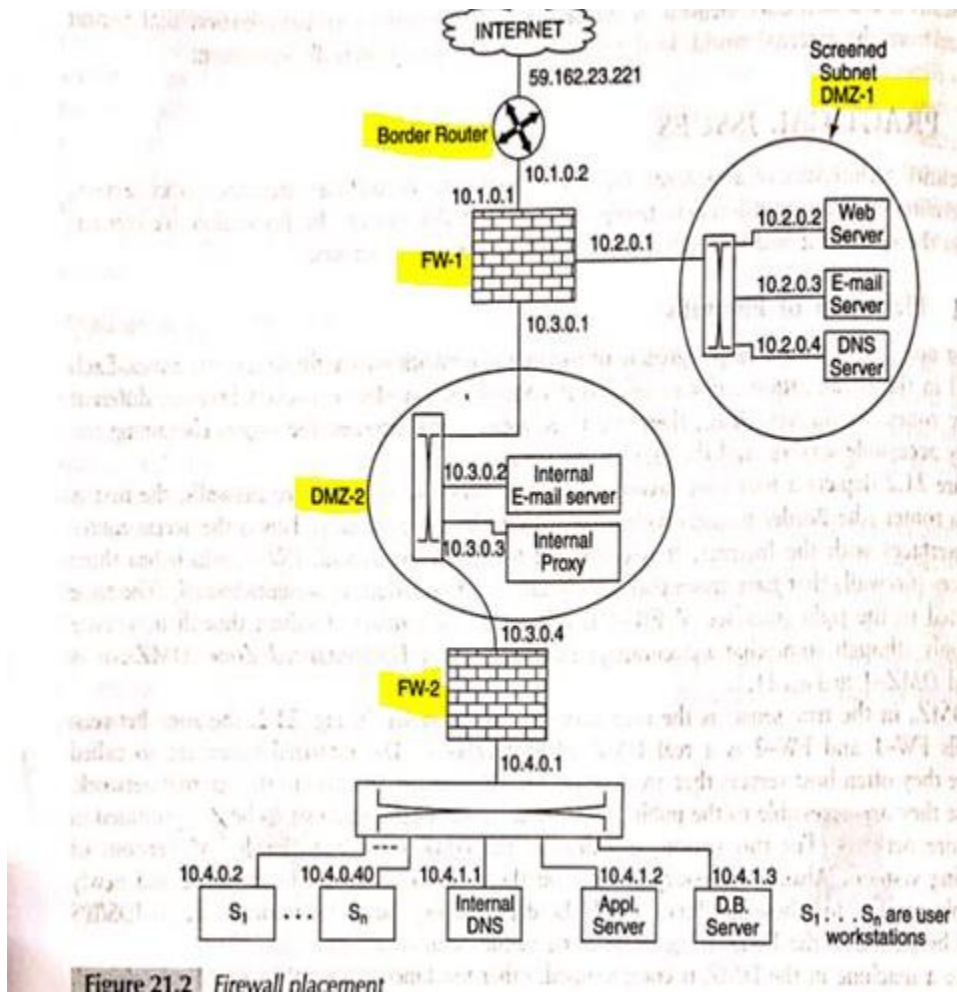
## 2.2 PRACTICAL ISSUES

- The security architecture of a medium size or large organization includes firewalls, proxy servers, VPN terminators, and intrusion detection/prevention (IDS/IPS) devices.

### 2.2.1 Placement of Firewalls

- We first note that firewalls help segregate or isolate the network into *multiple security zones*.
- Each firewall in the organization enforces rules that control the transfer of packets between different security zones.
- At the very least, there are three zones —

1. the Internet,
  2. the region containing the publicly accessible servers,
  3. and the internal network.
- Figure 21.2 depicts a four-zone layout using three firewalls.



- Of the three firewalls, the first is really a router (the Border Router) with some packet-filtering capability.
- This is the access router interfaces with the Internet.
- It is connected to a stateful firewall, FW-1, which has three interfaces (firewalls that have more than two interfaces are referred to as multi-homed).
- The zone connected to the right interface of FW-1 is referred to as *a screened subnet* though it is more commonly referred to as a De-Militarized Zone (DMZ). It is labelled DMZ-1 in Fig. 21.2. A DMZ, in the true sense, is the area between two firewalls.
- In Fig. 21.2, the zone between firewalls FW-1 and FW-2 is a real DMZ labelled *DMZ-2*.

- Demilitarized zones are so called because they often host servers that are accessible to the Internet and also to the internal network.
- Because they are accessible to the public, they are the most likely machines to be compromised in the entire network.
- Once a machine in the DMZ is compromised, other machines in the DMZ could get infected.
- DMZ-1 contains the publicly accessible servers.
- These include the web server, the external e-mail server, and the DNS server. All incoming mail from the Internet is received by this e-mail server, which checks for virus signatures and spam mail.
- The DNS server resolves names of publicly accessible servers. However, care should be taken to ensure that it does not contain address records of any of the internal machines. DMZ-2 contains the internal e-mail server. This is the server that hosts the mailboxes of the company employees. It handles the sending and receiving of all mail between internal parties. It Periodically establishes a connection to the external mail server (in DMZ-1) to retrieve all incoming mail.
- Outgoing mail (from the internal network to the Internet) can be handled in several ways. The internal mail server can set up an SMTP connection to a remote mail server to transfer mail.
- Alternatively, it can connect to the external mail server (in DMZ-1) and use it to relay all outgoing mail.
- DMZ-2 also contains an Internet proxy server.
- All internal users who wish to access external webpages connect to the proxy.
- The proxy authenticates the internal user and decides whether a page can be accessed (different restrictions might apply to different classes of users).
- The proxy scans incoming webpages for virus signatures and objectionable content. Finally, the proxy also performs caching of webpages.
- The internal network contains application servers, database servers, and user workstations.
- It also has an internal DNS server. This DNS server is different from the external DNS server in that it provides mappings between the domain names of the internal machines and their IP addresses.
- The internal machines all have private addresses. It is neither necessary nor desirable for third parties on the Internet to be aware of the private addresses of the internal machines. Hence, this DNS server is placed in the internal network.
- A feature of the security architecture in Fig. 21.2 is that services such as DNS and e-mail are *split*; that is, there is an internal DNS server as well as an external one.
- Likewise, there is an internal e-mail server and an external one.
- Generally, no external connection should be allowed to the internal servers.
- Connections in the reverse direction from the internal servers to hosts on the Internet should either be forbidden or severely restricted.

## 2.2.2 Firewall Configuration

- In order to create a firewall ruleset, we need to identify all the possible authorized connections that might be set up between pairs of machines in two different zones adjacent to the firewall.
- We first present a simplified version of the ruleset for firewall FW-2 (Table 21.2).
- Table 21.2 Simplified ruleset for firewall, FW-2

**Table 21.2** Simplified ruleset for firewall, FW-2

No.	From IP Addr.	From Port	To IP Addr.	To Port	Protocol	Action
1	*	*	Internal	*	*	Drop
2	User	*	Int_Mail_S	25	SMTP	Accept
3	User	*	Proxy	80	HTTP	Accept
4	*	*	DMZ-2	*	*	Drop

\*Wildcard

- The first rule states that **no machine** from any other security zone is permitted to establish a TCP connection to any internal machine.
- Rules 2-4 assert that, other than connections from internal stations to the internal mail server (on port 25) and web proxy (on port 80), no other connections are permitted to DMZ-1, DMZ-2, or the Internet.
- Table 21.3 shows the ruleset for firewall FW-1.

**Table 21.3** Simplified ruleset for firewall, FW-1

No.	From IP Addr.	From Port	To IP Addr.	To Port	Protocol	Action
1	*	*	DMZ-2	*	*	Drop
2	Int_Mail_S	*	Ext_Mail_S	25	SMTP	Accept
3	Internet	*	Ext_Mail_S	25	SMTP	Accept
4	Internet	*	Web_S	80	HTTP	Accept
5	Internet	*	DNS-S	53	UDP	Accept
6	*	*	DMZ-1	*	*	Drop
7	Proxy	*	Internet	80	HTTP	Accept
8	Ext_mail_S	*	Internet	25	SMTP	Accept
9	*	*	Internet	*	*	Drop

Rule 1 in Table 21.3 states that no TCP connection is to be established to any machine in DMZ-2 from any machine in DMZ-1 or the Internet.

Rule 2 states that the external mail server can accept connections from the internal mail server to receive incoming mail or to send outgoing mail.

Rule 3 allows connection to the external mail server from mail server on the internet to deposit incoming mail.

Rule 4 and 5 permit connections from the internet to the organizations web server and external DNS server, respectively.

Rule 6 states that no other connection may be set up to any machines in DMZ-1 for any other purpose.

Rule 7 and 8:the internet proxy in DMZ-2 and external mail server are permitted to make connections to machines on the internet to access webpages and to send outgoing mail.

Rule 9: confirms that no other connection from the organizations machine to the internet for any other purpose is allowed.

SVIT

---

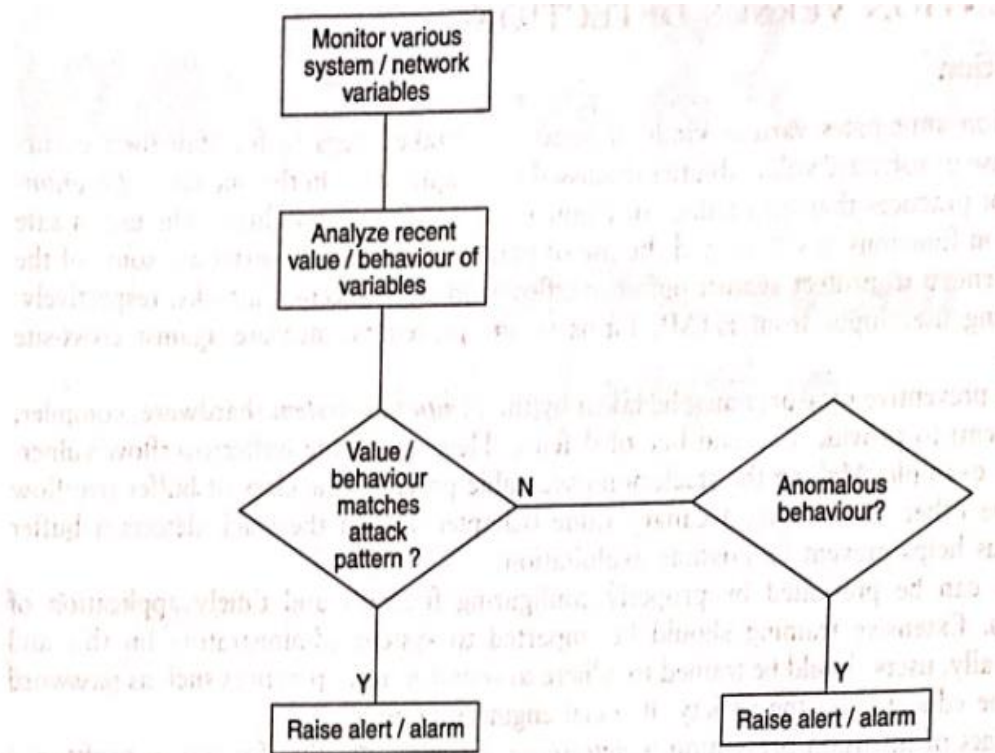
## MODULE 4

### *Viruses, Worms and Other Malware, Intrusion Prevention and Detection*

## INTRUSION PREVENTION AND DETECTION

### 22.1 Introduction

- **Definition** :An intrusion is the *act of gaining* unauthorized access to a system so as to cause loss or harm.
- Examples of intrusions include the following:
  - ✓ **Unauthorized login** to a system by illegally acquiring a password (through, for example, a password guessing attack). –
  - ✓ **Worm infections** that use the system as a launch pad to spread and infect other machines.
  - ✓ **Injection of spyware** that passively monitors the activities of the user and relays this information back to the attacker (over the Internet, for- example).
  - ✓ Flooding the host with **spurious connection requests** that attempt to exhaust the target's resources — processing power, memory, or communication bandwidth.
- Two ways of handling intrusions are **intrusion prevention and intrusion detection**.



**Figure 22.1** Tasks performed by an IDS

---

---

## 22.2 Prevention Versus Detection

<u>Prevention</u>	<u>Detection</u>
<ul style="list-style-type: none"><li>➤ Intrusion prevention anticipates various kinds of attacks and takes steps to forestall their occurrence.</li><li>➤ On the one hand, programmers should adopt practices that help reduce or eliminate software vulnerabilities.</li><li>➤ The use of safe string manipulation functions in C/C++ and the use of parameterized SQL queries are some of the practices recommended to protect against buffer overflow and SQL injection attacks, respectively.</li><li>➤ Likewise, sanitizing user input from HTML forms is one preventive measure against cross-site scripting attacks. Another set of preventive measures may be taken by the computing system (hardware, compiler, or operating system) to provide a second line of defence.</li><li>➤ Extensive training should be imparted to system administrators on this and related tasks.</li><li>➤ Finally, users should be trained to adhere to sound security practices such as password protection and be educated on the variety of social engineering attacks.</li><li>➤ One final aspect of intrusion prevention is deterrence. Hacking, whether for fun or profit, is a criminal offence.</li></ul>	<ul style="list-style-type: none"><li>➤ An intrusion detection system (IDS) (Fig. 22.1) performs the following three tasks:<ul style="list-style-type: none"><li>➤ First, it monitors "events of interest" occurring in the target system or in the network.</li><li>➤ An event of interest may be a system call (a call made to the operating system) to, for example, open a file containing sensitive data.</li><li>➤ Another event of interest may be the attempted establishment of a TCP connection from a specific IP address to a certain port. 2.</li></ul></li><li>➤ An IDS generates a large amount of data which it then analyzes and converts into valuable information to be used by system administrators.</li><li>➤ These are examples of thresholds and parameters set by a human.</li><li>➤ On the other hand, it would be highly desirable if the IDS were capable of learning what is normal behaviour, detecting anomalous events when they occur, and flagging such events.</li><li>➤ There are a number of key questions related to IDS functioning and deployment:<ul style="list-style-type: none"><li>➤ What are the variables that the IDS should monitor?</li><li>➤ When should an alert be raised? When should an alarm be sounded?</li><li>➤ Where should the IDS be placed?</li></ul></li></ul>

### 22.2.3 Case Study: Unauthorized User Logins

- To prevent unauthorized logins owing to compromised passwords, the following should be adhered to:
  1. A password should be at least **eight-characters** long, hard to guess, and include at least one non-alphanumeric character.
  2. A password should be changed at least *once in two months*.



3. Passwords should be *stored securely* (not written on sticker pads) and should not be communicated to friends, relatives, and co-workers.

4. After three consecutive *unsuccessful attempts* to a specific account, the system should be designed to disable all further log-in attempts for the next 20 minutes.

- Rules 1 and 2 must be enforced by the system.
- Rule 3 involves the user alone,
- rule 4 involves the system alone.
- These rules are all measures intended to prevent intrusion.
- As a further preventive measure, a high-security organization may mandate two-factor authentication — *passwords in conjunction with biometrics*.
- In addition to prevention, an IDS may also be deployed to monitor suspicious log-ins.

## 22.3 TYPES OF INTRUSION DETECTION SYSTEMS

- A real-world IDS monitors and mines hundreds of variables for interesting patterns.
- Table 22.1 shows a sample of variables together with a condition that may trigger an alert.
- Some of the variables are mere bit patterns in the packet header or the packet payload.
- Other variables are counts of a certain occurrence within a time interval.
- We next classify intrusion detection systems based on their functionality.

1) **Anomaly versus signature based IDS**

2) **Host based versus network based IDS**

**Table 22.1** Events of interest to an IDS

Variable monitored	Event of interest	Possible attack
No. of accesses to specific file	Tenfold increase over norm	DoS attack or flash event
Login frequency to particular account	Unusually high	Attempted break-in
No. of distinct source IP addresses of arriving packets	Very high	Worm attack
Ratio of ARP request packets to ARP response packets	$\gg 1$	Network scan to identify local active hosts
Ratio of TCP SYN packets to TCP FIN packets	$\gg 1$	Possible DoS attack
Percentage of half-open TCP connections	Sudden surge	Possible DoS/DDoS attack
TCP header flags	Invalid combination	Port scan, OS fingerprinting
TCP connection establishment	Unused destination port	Attempt to find which services are open
Payload of incoming packet	Specific bit sequence present	Specific worm attack
O.S. calls	Particular sequence of calls	Specific virus attack

**Table 22.1** Events of interest to an OS

### 22.3.1 Anomaly versus Signature-Based IDS

AnomalyBased IDS	Signature-Based IDS
<ul style="list-style-type: none"> <li>➤ Anomaly based intrusion detection involves making a determination whether the <i>behaviour of the system is a statistically significant departure from normal.</i></li> <li>➤ The IDS will have to learn, over time, what constitutes normal activity, usage, and behaviour.</li> <li>➤ The first six conditions in Table 22.1 are examples of what an <i>anomaly based IDS would monitor.</i></li> <li>➤ Consider monitoring the number of TCP SYN packets (with the <b>SYN flag set</b>) and FIN Packets (with the <b>FIN flag set</b>) in each successive 10-second interval.</li> <li>➤ A disproportionate number of SYN packets vis-a-vis FIN packets indicate several half-open TCP connections and possibly the onset of a <i>SYN flooding attack.</i></li> </ul>	<ul style="list-style-type: none"> <li>➤ <i>Signature-based intrusion detection</i> (also called <i>misuse detection</i>) works by identifying specific Patterns of events or behaviour that indicate or accompany an attack.</li> <li>➤ Each such pattern is called a <i>signature.</i></li> <li>➤ A signature-based IDS maintains a database of known <i>signatures.</i></li> <li>➤ It attempts to obtain a match between the <i>currently observed behaviour of the system and an entry in this database.</i></li> <li>➤ A real world signature-based IDS will have thousands of attack signatures against which to compare.</li> <li>➤ <i>An example of an attack signature is a specific bit sequence in a worm payload.</i></li> <li>➤ In a signature-based IDS it is the presence of a specific signature that raises an alert.</li> <li>➤ On the other hand, it is possible that a spread of the worm has caused much network traffic congestion and greatly increased CPU utilization on infected machines.</li> </ul>

### 22.3.2 Host-based versus Network-based IDS

Network-based IDS	Host-based IDS
<ul style="list-style-type: none"> <li>➤ An IDS that captures information about packets flowing through the network is referred to as <i>network-based IDS.</i></li> <li>➤ For reasons of performance, it is common to have stand-alone appliances that perform network-based intrusion detection. These typically run only the IDS and are hence not vulnerable to various worm and virus attacks.</li> </ul>	<ul style="list-style-type: none"> <li>➤ A host-based IDS is typically implemented in <i>software and resides on top of the host's operating system.</i></li> <li>➤ Its main job is to monitor the internal behaviour of the host such as the <i>sequence of system calls made, the files accessed,</i> etc.</li> <li>➤ For this purpose, it makes use of <i>system logs, application logs, and operating</i></li> </ul>

<ul style="list-style-type: none"> <li>➤ They may be <i>deployed at multiple points</i> in a large organization.</li> </ul>	<p><i>system audit trails</i> to identify events related to an intrusion.</p> <ul style="list-style-type: none"> <li>➤ <i>Operating system logs</i>, for example, keep track of when <i>users log in</i>, the number of <i>unsuccessful login attempts</i>, <i>the commands executed</i>, <i>network connections made</i>, etc.</li> <li>➤ Application logs keep track of which files have been opened or which registry keys have been accessed during the run of an application.</li> <li>➤ File system integrity checkers, for example, compute a cryptographic hash on the contents of each file. They detect file changes by comparing the computed hash of a file to its stored hash.</li> </ul>
---	--

- Two desirable features of an IDS are *speed and accuracy*.
- Speed is especially important in *fast-spreading Internet worms*, for example.
- *Early worm detection* and an early response mechanism such as automated system shutdown can help reduce the number of infected 'machines. The IDS should be able to detect every instance of an intrusion.
- An undetected intrusion is referred to as a *false negative*.

## 22.4 DDoS ATTACK PREVENTION and DETECTION

### 22.4.1 DDoS Prevention

- 1) Preventive Measures At The Host
- 2) Preventive Measures Inside The Network

#### Preventive Measures at The Host

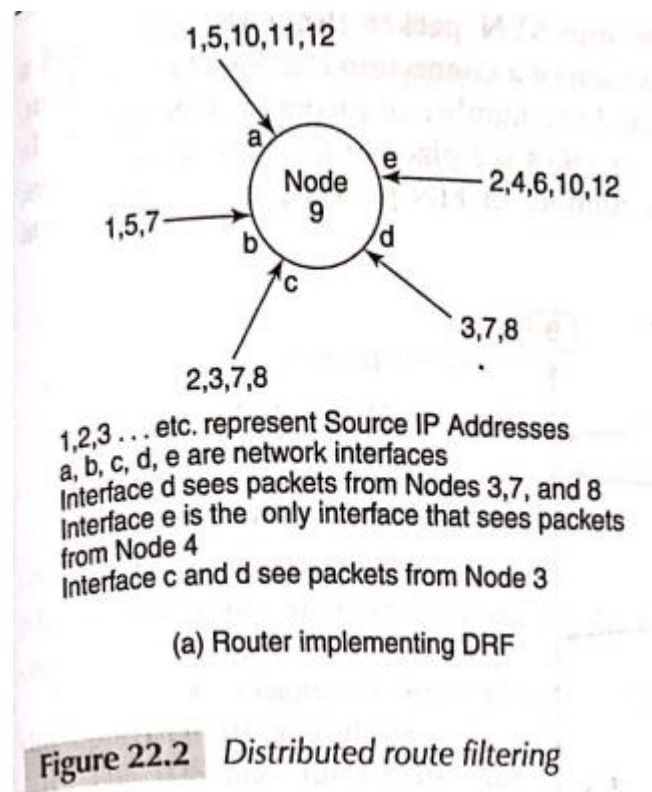
- One possible way of handling **SYN** attacks is to drop requests for TCP connections.
- But this could result in collateral damage if the victim is unable to distinguish between SYN packets that are part of the attack and those from its legitimate clients.
  1. One way to reduce collateral damage is to categorize IP addresses as "almost certainly genuine", "*probably spoofed*", etc.
  2. The "*almost certainly genuine*" addresses are those with whom normal connections were established and terminated in the past.
  3. Under rapidly increasing load, packets with *unfamiliar source addresses* are discarded with high probability.

- 
- 
- Another strategy under high-load conditions is to allocate a full buffer of about 300 bytes for a given TCP connection request only upon completion of the three-way handshake.
    1. While the connection is still half-open, minimal information about it is stored in a hash table called the SYN cache.
    2. This information includes the TCP sequence numbers and source/destination addresses and ports.
    3. An alternative to the SYN cache is the SYN cookie, which stores no state information at all for each half-open connection.
    4. Instead, the responding machine places a cookie within the *Sequence Number field of the second handshake message*.
    5. The cookie is computed as a hash function of the source address, destination address, source port, destination port, and a secret.
    6. The initiator of the connection dispatches the cookie it just received in its ACK message (third message of the three-way handshake).
    7. Upon receiving the ACK, the responder re-calculates the cookie and verifies that it matches the value enclosed in the received ACK.
    8. Only then does it reserve buffer space for the connection.
    9. If the source IP address in the first message of the handshake were spoofed, the cookie in second message would not be received by the initiator but by the machine corresponding to the spoofed IP, address.
    10. The initiator would not be able to complete the three-way handshake since it does not know the cookie value. Hence, its connection request would not be granted buffer space.

### Preventive Measures inside the Network

- An intuitively appealing approach to frustrating DDoS attacks is to implement measures closer to the source of the attack.
- One such measure is *egress filtering*.
- **Attack:** Most DDoS attack packets use *spoofed source IP addresses*.
- Address spoofing is employed to confuse cyber sleuths making it hard to pinpoint the true source of the attack.
- The perpetrator hopes to continue the attack for as long as desired and perhaps even resume it at a later point without being traced.
- **Solution:** The *egress router is the last router* encountered by any packet generated inside the network before it exits that network and enters the Internet.
- Let A be the set of all externally visible IP addresses within the network (behind the egress router). The egress router examines the source address of each packet leaving it. If the address does not match any address in A, it drops the packet. By thus detecting and filtering spoofed packets' it helps prevent DDoS attacks.
- The idea of *egress filtering has been extended to routers* in the core of the Internet.

- A filter, on the other hand, uses the packet's source address to make a decision on whether or not to discard the packet.
- To implement **Distributed Route Filtering (DRF)**, a filter maintains, for each of its interfaces, the set of all **source addresses** from which packets arrive en route to some destination.
- The router uses **BGP routing information** to obtain the **latest mapping** between each of its interfaces and the subset of source addresses using that interface.
- The filtering decision is straightforward — if a packet with source IP address = S arrives via an interface that it should not have, that packet is assumed to be spoofed and is hence discarded.
- Figure 22.2(a) shows an example of a router implementing DRF.

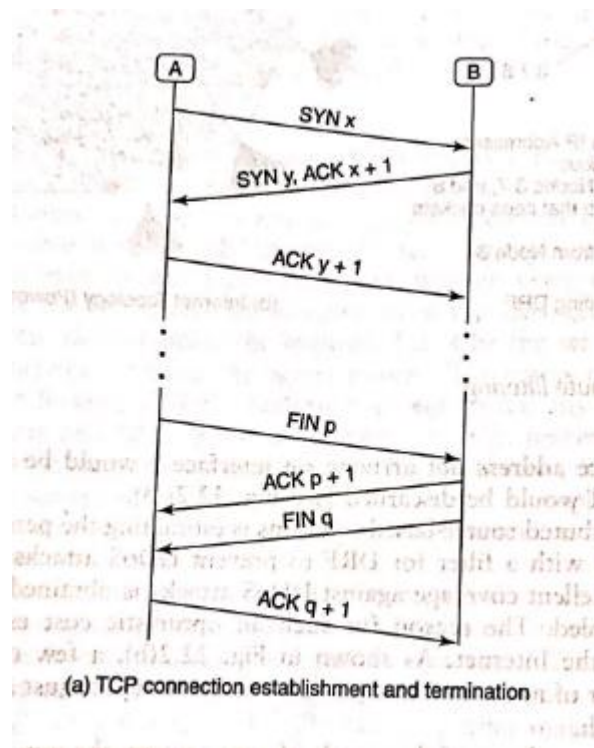


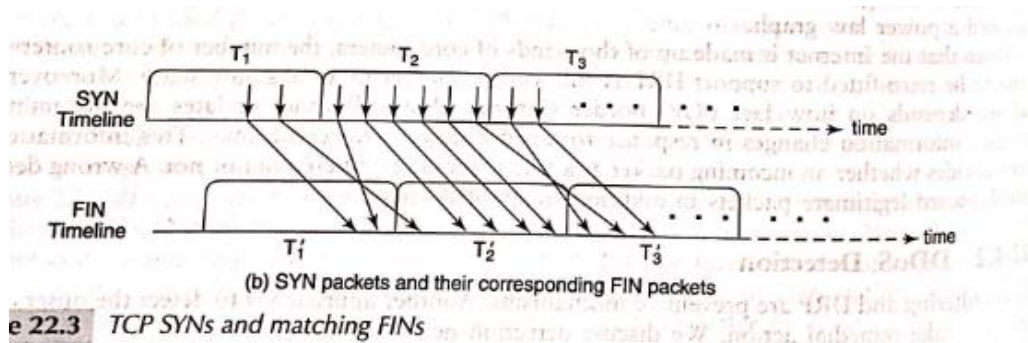
- Each of its interfaces is marked with the source addresses that use that interface en route to some destination.
- Note that packets from the same source may enter the router through different interfaces.
- For example, packets from source address 7 may arrive through interfaces b, c, or d.
- In the simplest implementation of the filter, the **router checks whether a packet has arrived on one of its "acceptable" interfaces based only on the packet's source IP address.**

- For example, a packet bearing source address = 7 arriving on interface c would be forwarded. However, another packet with the same source address but arriving on interface e would be suspected of having a spoofed source address and would be discarded [see Fig. 22.2

### 22.4.2 DDoS Detection

- Egress filtering and DRF are preventive mechanisms.
- Another approach is to detect the onset of DoS and then take remedial action.
- In a SYN flood attack, the victim sees a disproportionate number of SYN packets compared to FIN packets.
- By a SYN packet, we mean any incoming packet with the SYN flag set.
- A FIN packet is sent by the side that wishes to terminate the TCP connection.
- If the other party agrees to termination, it responds with its own FIN packet. Thus, SYN and FIN packets usually occur in pairs.





- Figure 22.3(b) shows two horizontal timelines — the top line shows the times of SYN packet arrivals.
- The bottom line shows the corresponding FIN arrivals.
- Time is slotted into fixed-length observation intervals, "  $T_1, \dots$  , during which we record the number of SYN arrivals.
- The corresponding observation intervals for FINs,  $T'_1, T'_2, \dots$  are shifted to the right by the average duration, of a TCP connection.
- To construct an anomaly detection system, we define the following variables as
- **$S_i$  = # of SYN packet arrivals in the  $i$ -th observation interval**
- **$F_i$  = # of FIN packet arrivals in the  $i$ -th observation interval**
- **$D_i$  = normalized difference between # of SYN and FIN packets in the  $i$ -th observation interval, i.e.,**

$$D_i = \frac{S_i - F_i}{F_i}$$

$\tau \equiv$  threshold for detection

Consider the *time series*,

$$D_1, D_2, D_3, \dots$$

- The different algorithms that attempt to detect the onset of a SYN Flood Attack by monitoring the above series.
- 1. Algorithm 1. Raise an alert if the most recently computed detection variable  $D_i$  exceeds the threshold, i.e.,  $D_i > \tau$**
- Figure 22.4(a) shows  $D$  versus time with the threshold set at  $\tau = 90$ .
- Some of the problems with this approach are as follows:
  - (i) The IDS may raise many **false alarms** since it bases its decision on point values.
  - (ii) A modest spike in  $D$  at just one point is very unlikely to result in memory exhaustion but it does cause the IDS to raise an alarm.
  - (iii) The *cumulative* effect of the attack packets across the interval will cripple the system but this algorithm will not raise an alarm.

---

2. **Algorithm 2 : Raise an alert if the "smoothed average" of the previous values of D exceeds the threshold.**

- This approach uses the well-known technique of *exponential smoothing*.
- The decision variable at the end of the i-th observation interval is the smoothed average,  $S_i$  computed using :

$$S_i = \alpha D_i + (1 - \alpha) S_{i-1} \quad 0 < \alpha < 1 \text{ and } S_0 = 0$$

$$S_i = \alpha D_i + \alpha(1 - \alpha) D_{i-1} + \alpha(1 - \alpha)^2 D_{i-2} \dots$$

3. **Algorithm 3. Define a modified cumulative sum of previous values of D. Raise an alert if this value exceeds a threshold.**

- During normal operation, the number of FINs will balance out the number of SYNs and hence  $D_i$  will be close to 0.
- Let  $u$  be an upper bound on the mean of  $D_i$  during normal operations.
- Let  $D_i'$  be a shifted version of  $D_i$ , i.e.,
- $D_i' = D_i - u$ .
- The decision variable,  $M_i$  used here is defined as
- $M_i = (M_{i-1} + D_i')$  where  $M_0 = 0$ ;

### 22.4.3 IP Traceback

- There are two principal approaches to IP traceback :
- **packet marking**: the packet keeps track of the routers it has visited
- **packet logging**: each router keeps track of the packets passing through it.
- hybrid approaches using a combination of packet marking and packet logging have been proposed.

#### Probabilistic Packet Marking

- Consider, for a moment that every intermediate router were to **append its 32-bit IP address** to each packet it forwards.
- A packet on the Internet traverses about 10 hops on the average, so an extra 40 bytes would be needed to keep track of its path from source to destination.
- This is an unacceptable per-packet overhead.
- Instead, existing but infrequently used fields in the IP header are used to keep track of the routers visited.
- The IP header has a **16-bit ID field**.
- This field provides support for packet fragmentation and re-assembly.
- Different networks have different restrictions on the size of the datagrams they can carry.



- 
- 
- They may split a datagram into two or more fragments and send each fragment separately.
  - The router at the destination end has the responsibility for reassembling the fragments
  - To create the original packet.
  - All the fragments carry the same number in their ID fields, so they can be identified for re-assembly.
  - On the assumption that the ID field is often unused, traceback schemes employing PPM use the ID field to store partial information on intermediate routers.
  - But, given that the length of each IP address is 4 bytes, how can a packet store router address information in a 16-bit ID field?
  - The answer lies in computing a global fingerprint for each router — this is, say, 16 or fewer bits . of the hash of a router's IP address.
  - An intermediate router writes its fingerprint value into the ID field of a packet with probability  $p$ .
  - Note that it could over-write a previously written fingerprint of a router closer to the source of the attack.
  - To identify the perpetrator of the attack, the ingress router at the victim end will need to collect a sufficient number of packets that are all part of the same flooding attack.
  - We assume that each ingress router has a map of all upstream routers from it.

#### *Packet Logging*

- ***Each router attempts to keep track of every packet that passes through it.***
- Packet logging makes use of the idea of a packet fingerprint or digest.
- This is computed using a well-designed ***hash function*** — one that distributes the hash values uniformly across all possible hash inputs.
- An interesting feature of packet logging is that it can help track even a single rogue packet.
- First, assume that each router stores each packet received by it in the last 5 minutes.
- Suppose the victim wishes to obtain the exact path followed by a packet received by it.
- The idea is that the victim's ingress router, A, queries each of its adjacent routers whether they have seen the packet.

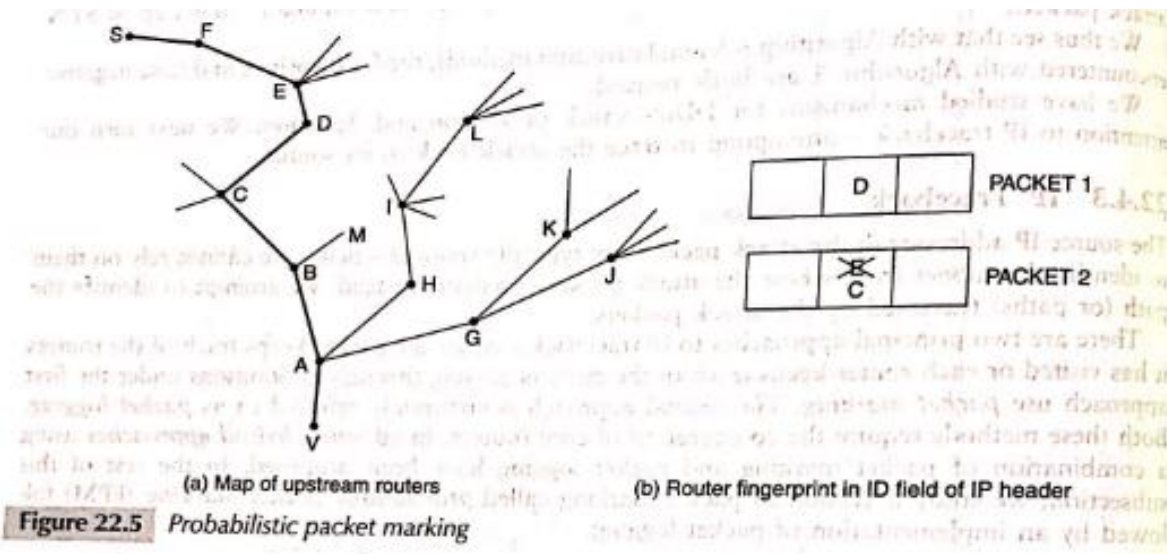
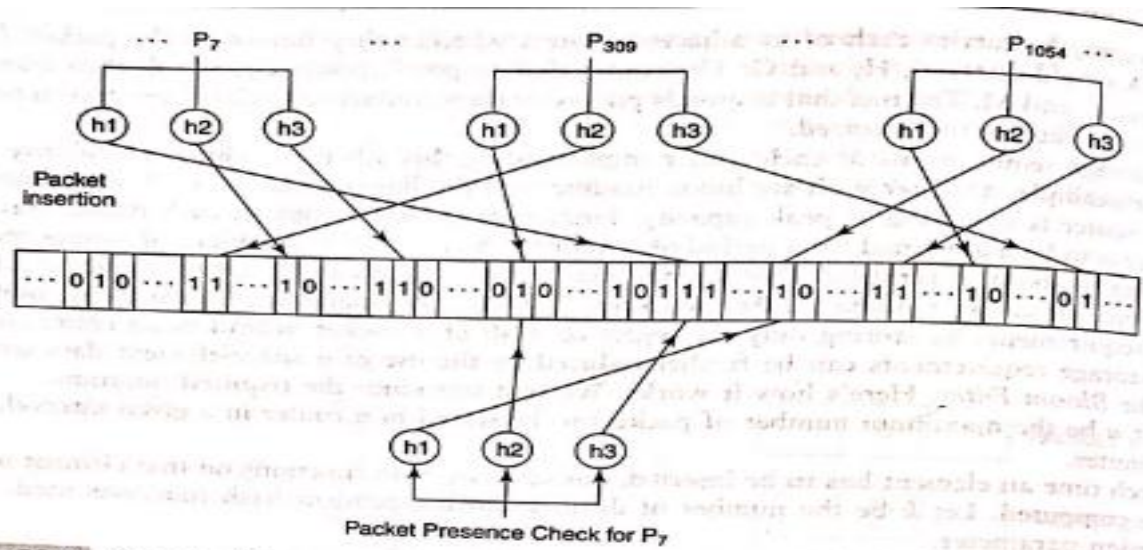


Figure 22.5 Probabilistic packet marking

- In Fig. 22.5(a), A would query B, H, and G.
- The router that responds positively, say B, then queries its neighbours, C and M.
- The one that responds positively then contacts its neighbours and so on until the source of the packet is traced.
- The storage requirements can be further reduced by the use of a space-efficient data structure called the Bloom Filter.
- Let  $n$  be the maximum number of packets to be stored in a router in a given interval, say 7 minutes.
- Each time an element has to be inserted, one or more hash functions on that element need to be computed.
- Let  $k$  be the number of distinct and independent hash functions used.  $k$  is a design parameter.
- The output of each hash function returns a  $w$ -bit quantity.
- The Bloom Filter is basically a bit array.
- Let  $m = 2^w$  be the size of this array.
- **Packet "Insertion."**: When a packet enters the router, the  $k$  hashes are computed on its content.
- To speed up the computation, the hashes are only computed on the invariant parts of the IP header and a small part of the payload, say 10 bytes.
- Suppose the  $k$  hash computations yield the values  $i_1, i_2, i_3, \dots, i_k$ .
- These  $k$  hash outputs are used as indices into the bit array.
- To "insert" a packet, the bits in those positions are all set to 1. (If one or more of them were already set, they remain set.)
- **Packet Presence Check:** To check if a packet,  $P$ , is present in the Bloom Filter, compute the  $k$  hashes on it as done during packet insertion.

- Suppose the  $k$  hash computations yield the values  $i_1, i_2, i_3, \dots, i_k$ . Then, check whether each of the elements of the Bloom Filter are set. If even one of these elements = 0,  $P$  has not been encountered by this router.
- We next derive an expression for the probability of a false positive.



**Figure 22.6** Illustrating false positive in a Bloom Filter

Probability [a packet hashes to  $i_1$ ] =  $\frac{1}{m}$

Probability [a packet does not hash to  $i_1$ ]  
 =  $\left(1 - \frac{1}{m}\right)$

Probability [none of the  $n$  packets hash to  $i_1$  with any of the  $k$  hash functions]  
 =  $\left(1 - \frac{1}{m}\right)^{kn}$

Probability [at least one of the  $n$  packets hashes to  $i_1$  with at least one of the  $k$  hash functions]  
 =  $1 - \left(1 - \frac{1}{m}\right)^{kn}$

---

---

and at least one of the  $n$  packets hash to  $i_k$  with at least one of the  $k$  hash functions]

$$= \left( 1 - \left( 1 - \frac{1}{m} \right)^{kn} \right)^k$$

It turns out that a reasonably acceptable false positive probability of 1% is achievable with  $k = 3$  and  $m = 12 \times n$ . This translates to a storage cost of only 12 bits per packet at the expense of performing three hash computations per packet. This is considerably less than storing each IP datagram (about 500 bytes on average) or even storing just a 32-bit hash of a packet.

## Virus, Worms and Malware

### 19.2 VIRUS AND WORM FEATURES

#### 19.2.1 Virus Characteristics

- When a virus-infected program is run, the virus code is executed first.
- One of the first tasks of virus code is to seek other programs not yet infected and then pass on the infection to one or more of them.
- A truly malicious virus may then perform actions such as deleting certain files.
- An innocuous virus may attempt something benign like printing a "hello world" message.
- Execution of the virus code is usually followed by execution of the host's original program.
- All the virus code need not be located at the start of the infected file.
- In some cases, virus code is both prepended and appended to the host file.
- Virus code could be split into several segments and interspersed throughout the infected file using JUMP statements at the end of each virus segment.
- In most of these cases, the size of the infected program is larger than the original host program. This helps anti-virus software to detect infected code.
- To evade detection, some viruses modify the file service interrupt handler that returns attributes of files. By so doing, the service handler may be programmed to return the uninfected length of the file.
- Another technique is to use compression so that the length of an infected file remains the same as the length of its original version. The virus writer includes a compression routine in the viral code. To infect another file, the virus first compresses that file and then prepends the virus code to the compressed file.
- The infected file must be uncompressed just prior to execution.

- 
- 
- One of the characteristic features of many viruses is the set of system calls they make. System calls are used by application programs to request services of the operating system.
  - They are made to read/write files, spawn new processes, establish TCP connections, etc. Some viruses make calls to copy their own code to other files, create/modify entries in the Windows registry, or search for e-mail. Such "suspicious" calls are often used to distinguish malicious from benign code.

### 19.2.2 Worm Characteristics

- Classes and features
- Worms are most commonly classified based on their vector of propagation.
- The main categories include:
  1. *Internet scanning worms*
  2. *E mail worms*
  3. *P2P worms*
  4. *Web worms*
  5. *Mobile worms*
- *Enhanced Targeting*
  - ✓ The most important attribute of a Worm is that it spreads its infection to other computers.
  - ✓ Many target selection strategies have been proposed and implemented.
  - ✓ Worms that spread through e-mail, for example, have an easy way to figure out their targets.
  - ✓ All they need to do is look into their victim's mailbox or e-mail address book to find a set of targets.
  - ✓ A mobile worm obtains phone numbers of its potential victims from the phone book in the cellphone hosting the worm.
  - ✓ Some web worms use search engines to harvest URLs of potentially vulnerable targets.
  - ✓ Internet scanning worms, on the other hand, scan the IP address space for vulnerable machines.
  - ✓ The most straightforward approach is random scanning — choosing IP addresses at random. This was adopted by Code Red Version-I. However, Code Red Version-II adopted localized scanning.
  - ✓ Over 80% of the time, it attempted to connect to victims with whom it shared the network address (most significant 8 or 16 bits of the IP address). This strategy was more successful since hosts in the same network are likely to be closer and be running the same software.
  - ✓ Worms like Nimda, unleashed in September 2001, spread aggressively thanks to its five different vectors of propagation. Propagation through HTTP and e-

---

---

mail were particularly successful in penetrating the perimeter of the enterprise. Once inside, it exploited the Windows file-sharing feature to spread within the enterprise.

➤ ***Enhanced Speed***

- ✓ To enhance the infection rate, some worms are designed to spawn multiple threads.
- ✓ Each thread is responsible for setting up connections to a different subset of hosts, thus increasing the rate at which infection is spread.
- ✓ Some worms reduce infection latency by targeting a buffer overflow vulnerability on an application that employs UDP rather than TCP.
- ✓ TCP connection establishment involves a three-way handshake and is time-consuming.
- ✓ UDP, by contrast, is connectionless.
- ✓ This sharply reduces infection latency.
- ✓ A steep increase in the number of infected machines at the very outset of a worm epidemic has a multiplicative effect on spreading rate.
- ✓ For this purpose, the attacker could create one or more hit-lists carrying addresses of several thousand vulnerable machines.
- ✓ The first worms to be let loose could carry one such list.
- ✓ As a worm infects each new machine, it splits its list between itself and the machine it has just infected.
- ✓ Given that most of the machines on the hit-lists are vulnerable, the worm spreads rapidly during the initial stage of the epidemic. Thereafter, the infected machines could spread the infection using random scanning or some other spreading method.

➤ ***Enhanced Capabilities***

- ✓ Most worms (and viruses) have unique and distinct signatures — a pattern of bits, usually assembly language code, which appears in all instances of the worm.
- ✓ Worm and virus signatures are the key to detecting them. However, there are sophisticated code obfuscation techniques to evade detection.
- ✓ One such technique is the use of encryption for disguising worm code.
- ✓ Different instances of the worm may use different keys for encryption. Thus, they might fail to match any existing worm signatures. Such worms are said to be polymorphic.
- ✓ A polymorphic worm would have to be decrypted before being executed. This suggests that a decryptor routine "in the clear" would have to be part of the worm code.
- ✓ Decryptors themselves may be very simple, involving XOR operations or trivial shift-based substitutions. However, detecting a worm on the assumption that the decryptor routine is invariant would not always succeed.

- ✓ Figure 19.1 shows two versions of assembly code that look different but perform the same function.
- ✓ The second version is inefficient with spurious instructions.
- ✓ The second version also has a spurious branch instruction to confuse worm code detection software that relies on control flow analysis.
- ✓ Worms that have multiple such versions with or without relying on encryption are referred to as metamorphic worms.

```

Assembly Pseudo-code
    if R5 > 0
        R4 ← R1 + R2
    else
        R4 ← 4 × R2 + 3 × R3

Assembly Code: Version 1
    First:  CMP R5, #0
           BLE Second
           ADD R1, R2, R4
           BRA Finish
    Second: ADD R2, R3, R4
           SLA R4, #2, R4
           SUB R4, R3, R4
    Finish: ...

Assembly Code: Version 2
           XOR R6, R6, 0
           ADD R5, R6, R6
           SUB R6, #0, R6
           BG First
    Second: ADD R2, R2, R4
           ADD R4, R4, R4
           ADD R4, R3, R4
           ADD R4, R3, R4
           ADD R4, R3, R4
           BNE Finish
           CMP R4, R4
           BE Finish
    First:  ADD R1, R2, R4
    Finish: ...

```

Figure 19.1 Polymorphic assembly language versions of same pseudo-code

➤ **Enhanced Destructive Power**

- ✓ It is estimated that worms such as Code Red and Nimda caused billions of dollars in damage.
- ✓ Analysts estimate costs based on lost productivity, clean-up costs, system downtime which affects business and revenues.
- ✓ Fast-spreading worms also caused severe network congestion problems disrupting normal Internet traffic and contributing to system dos time.
- ✓ Nevertheless, most worms thus far have been relatively benign.
- ✓ Some worms contributed atta packets to a DDoS attack or caused website defacement.
- ✓ The Witty worm which appeared in Mar 2004, however, was qualitatively different. It was the first worm to carry a destructive payload. deleted a random section of the victim's hard disk leading to a system crash

---

## 19.3 INTERNET SCANNING WORMS

- ✓ One characteristic of Internet scanning worms is that they are self-activated.
- ✓ The ability to spread without human intervention distinguishes them from most types of e-mail, P2P, and web worms.
- ✓ This category of worms is so called because they scan the Internet looking for vulnerable machines.
- ✓ The vulnerability could be a buffer overflow problem in a commonly used service provided by a particular version of an OS.
- ✓ The worm communicates with and delivers its malicious payload to the victim using standard transport protocols such as TCP or UDP.
- ✓ Once installed on the victim, it could erase local files, steal secrets, or deface webpages, but above all it seeks new victims to infect.

### 19.3.1 Case Studies: Code Red and Slammer

#### *Code Red*

- One of the best known examples of Internet scanning worms is the Code Red worm,
- It all started on June 18, 2001, when a buffer overflow vulnerability was discovered in the Microsoft IIS Web Server.
- A patch for this vulnerability was developed a few days later.
- It is estimated that there were several million IIS servers in active deployment.
- Even assuming that a large percentage of these were patched, that still left plenty of room for the spread of the worm, which was unleashed on July 12, 2001.
- The worm itself was carried in HTTP request messages targeted at IIS servers.
- The first version of the worm used a random number generator to generate new addresses of machines to infect. However, the same seed was used for the random number generator in every instance of the worm.

#### *Slammer*

- The SQL Slammer was launched on 25 January, 2003, and targeted a buffer overflow vulnerability on the Microsoft SQL server 2000.
- The worm sent packets on UDP port 1434 — the database software's resolution service.
- It used simple random scanning to propagate.
- Slammer's payload was a mere 384 bytes in length — far smaller than the 4 kb payload of Code Red. Also, UDP, being a connectionless protocol, there is no overhead of connection establishment.



---

## Worm Propagation Models

### Simple Epidemic Model

- The Simple Epidemic Model used to study the spread of infectious diseases among humans is an appropriate starting point.
- The model assumes that there are only two types of entities in the population.
- Either an individual is susceptible or he is infected.
- An infected individual can infect a susceptible person.
- Once infected, a person remains infected and does not recover.
- Let  $N$  be the size of the total population.

Let  $N$  be the size of the total population. Let  $I(t)$  be the number of infected individuals at time  $t$ . The number of susceptibles at time  $t$  is then  $N - I(t)$ .  $\beta$  is the initial infection rate, i.e., each infected person attempts to pass on the infection to  $\beta$  susceptibles in 1 time unit. The following differential equation captures the number of infected persons at time  $t$ .

$$dI = \beta I(t) \left(1 - \frac{I(t)}{N}\right) dt \quad (19.1)$$

or

$$\beta dt = \left( \frac{dI(t)}{I(t) \left(1 - \frac{I(t)}{N}\right)} \right) \quad (19.2)$$

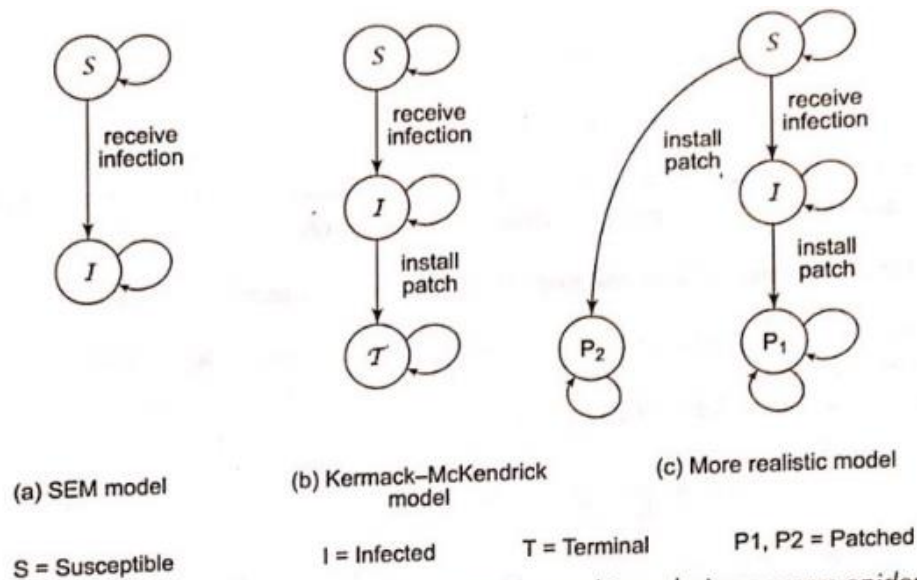
In an infinite population, each infected person infects  $\beta dt$  susceptibles in time interval  $dt$ . However, in a finite population of size  $N$ , the probability that the target of an infective is already infected is  $\frac{I(t)}{N}$ . Such targets do not add to the population of newly infected. The factor  $\left(1 - \frac{I(t)}{N}\right)$  in the above equations ensures that only previously uninfected entities are added to the count of the freshly infected in time interval  $dt$ .

Integrating both sides of Eq. (19.2) yields

$$I(t) = \frac{I_0 N}{I_0 + (N - I_0)e^{-\beta t}} \quad (19.3)$$

### Kermack—McKendrick Model

- The Kermack—McKendrick (K—M) model more accurately models the spread of human infectious disease by considering three (instead of two) categories of people:
  - those who are susceptible (state S)
  - those who are infectious (state I) and
  - those who are neither, i.e. individuals who are cured or those who have succumbed to the disease (terminal T).
- Initially, all individuals in the population are susceptible.
- It is possible to go from state S to I but not vice versa .



**Figure 19.3** State machine representation of vulnerable machines during a worm epidemic

## 19.4 Topological worms

### 19.4.1 Email worms

- Email worm propagates through infected e mail.
- The victim receives email that comes from trusted or familiar source .
- The victim sees an innocent text file attached to the email.
- In reality, the file named such as loveletter.text.vbs contains Visual Basic script.
- By clicking on this attachment, the embedded VB script executes, sending a copy of itself to every person in the victim's contact list.
- Many e-mail worms exploit the fact that documents created by certain word processors embed software macros in them.
- The macros execute when the document is opened
- For example, Melissa was a macro worm (or "macro virus") that propagated by sending copies of itself to the first 50 persons in the victim's address book.
- One of the best-known e-mail worms of more recent vintage is Sobig, which was let loose in 2003. It spread by communicating malicious c-mail or copying itself to an open network share.
- There were several versions of SoBig. One version could update itself by downloading code from certain websites.
- The URLs of these sites were contained in a file that itself was downloadable from geocities.com this site allows users to host their own free webpages besides providing tools in support of building dynamic webpages).
- Some of the malicious code received , installed a keystroke logger and stole passwords from its victims.

---

---

## 19.4.2 P2P Worms

- A P2P network is a massively distributed system of computers where each peer or node plays the role of both client and server.
- They are used principally for sharing files, which may contain songs, images, videos, etc.
- Each peer maintains within itself a shared folder of files that it is willing to share with others.
- Users do not download files from a central server but from their peers located across the globe.
- They are immensely popular as evidenced by the fact that a very large proportion of Internet traffic is comprised of P2P packets.
- To see how P2P worms spread, it is important to understand how a P2P network operates.
- Most P2P networks use an overlay network, which is a logical network of peers.
- Two peers are said to be neighbors at any given point of time if there is an active TCP connection between them.
- Here are potential ways in which P2P worms may spread:
  1. ***One of the simplest is for a malicious peer to respond positively to any query.***
    - ✓ If the requester then chooses to download the file from the malicious peer, the latter sends it an infected file whose name is changed to match that of the requested file.
    - ✓ The infected file contains a worm which passes on its infection to the requester.
    - ✓ Once infected, the requester mimics the behaviour of the malicious peer thus helping to propagate the infection.
    - ✓ Alternatively, various "popular" files stored in the shared folder of a peer may be infected.
    - ✓ When any of them is downloaded, the infection spreads to the shared folder of the requesting peer.
  2. ***Peers in a given P2P network run the same P2P protocol.***
    - ✓ There are usually few different implementations of this protocol leading to little software diversity.
    - ✓ An exploitable buffer overflow vulnerability in one popular implementation is a familiar starting point.
    - ✓ This, coupled with the fact that a peer maintains a list of neighbours, implies that a worm has ready targets and does not need to perform random scanning as in the case of Internet scanning worms.
    - ✓ The first type of worm is said to be passive since it propagates only when requested to download a file.
    - ✓ The second type of worm is active since it propagates on its own without receiving requests from its peers.

- 
- ✓ One aspect of P2P worms is that they may result in no apparent traffic anomaly, so an intrusion detection system monitoring network traffic is unlikely to raise an alert.

## 19.5 WEB WORMS AND CASE STUDY

- Web worms differ from malware such as the Internet scanning worms in several ways.
- Many web worms are executed in browsers which run on diverse hardware/OS platforms.
- Web worms are written in a high-level language making it easy to perform complex operations but difficult to execute low-level operations.
- On the other hand, many other worms are written in assembly language.
- One type of web worm is the XSS worm — so called since it exploits cross-site scripting vulnerabilities in web servers
- The first step in creating an XSS worm is to inject attack code into a vulnerable web server.
- When a user accesses the infected website through his/her browser, the malicious code (usually Javascript) is downloaded on to the browser.
- As in any XSS vulnerability, malicious code executes on the browser.
- Given that a key function of a worm is to propagate, the challenging question then is "How does an XSS worm propagate?"
- A partial answer to this question may be found through our next case study of the Samy worm.

### *XSS Worm Case Study*

- The XSS worm, Samy was unleashed in October 2005.
- Authored by SamyKamkar, it infected the social networking site, Myspace.
- Social networking sites typically allow users to create and edit their profiles (Fig. 19.5), which are stored on the site and are accessible to other members of the social networking group.
- A user profile may contain information about him including his hobbies, photographs, etc.
- A user profile also contains a list of the user's friends with hyperlinks to their profiles.
- Samy added a bunch of carefully crafted Javascript to his profile.
- When a visitor to Samy's website, say V1, downloaded Samy's profile on to his browser, the Javascript in Samy's profile executed.
- This caused Samy to be added as a friend in V1's profile and also to include the message "but most of all, Samy is my hero."
- " Within 20 hours of the first visit to Samy's profile, Samy had been added as a friend to more than a million user profiles.
- This rate of spread was even faster than that of Code Red.
- How did the worm spread and why did it spread so fast?

- 
- The malicious Javascript uploaded itself on to V1's profile on the MySpace server, thus infecting it.
  - This is done by an HTTP Post-request sent from the browser to the server. However, that would cause the screen to freeze between sending the request and receiving the HTTP response from the server.
  - To ensure that the viewer had a normal screen experience, Samy'sJavascript created an XMLHttpRequest object which was used to send the malicious Javascript to the Myspace server. Unlike the regular HTTP request, the message from an XMLHttpRequest object is asynchronous and runs in the background.

## 19.6 MOBILE MALWARE

### 19.6.1 Introduction

- New—generation smartphones combine the functionality of a cellphone and a lose-end PC.
- They may be used for storing confidential documents, communicating via e-mail/SMS/MMS, and taking photographs.
- They support feature-rich applications that run on top of a complete OS.
- The most common OS on smartphones is the Symbian followed by Windows Mobile, Linux, and recently the Mac OS X (on the iPhone).
- They provide a rich set of APIs to access the phone book and other files, send SMS/MMS messages, etc. Unfortunately, these very APIs can also be used by malware to, for example, read a confidential document on the smartphone and ship it to the attacker as an MMS attachment.

### 19.6.2 Bluetooth

- Bluetooth is both a communication technology and a protocol stack.
- As a communication technology,
  - ✓ It supports short-range wireless communication —
  - ✓ A maximum of between 10 and 100 meters between devices.
  - ✓ Bluetooth uses 2.4 ghz shortwave radio technology.
  - ✓ Bluetooth is a complex, multi-layered protocol.

#### *Discovery and User Authorization*

- Besides the vulnerabilities mentioned above, social engineering is a key factor in the spread of mobile malware,
- Many PC users do not hesitate to click hot links in their e-mail or open attachments even when the sender of the e-mail is unknown.
- This behavior carries over to the mobile world where many users have no hesitation in accepting a file from an unknown source.

- 
- 
- The combination of Bluetooth implementation vulnerabilities, rich feature set of the smartphone, and unthinking user behavior has exposed the smartphone to various strains of malware.
  - To investigate the spread of malware in smartphones, it is necessary to understand the basics of how smart phones exchange files using Bluetooth.
  - To discover other Bluetooth-enabled devices in its neighborhood, a device initiates an inquiry procedure, which includes broadcasting an inquiry request.
  - All devices in the range of the initiator that are in discoverable mode respond sending their bluetooth device address (BD\_ADDR).
  - This is a 48-bit MAC address — the first 24 bits identify the device manufacturer/model and the last 24 bits specify a particular instance of that model.
  - Bluetooth is a connection-oriented protocol.
  - A device, A, can set up a connection to any other device B. But to set up such a connection, A should know the BD\_ADDR of B.
  - One way of obtaining B's BD\_ADDR is through the discovery procedure.
  - A large percentage of users keep their phones in discoverable mode, so this is an attractive way of harvesting device addresses especially In crowded areas such as railway stations or malls. However, it should be noted that even with the phone in non-discoverable mode, there are a number of brute force techniques to extract its BD\_ADDR.
  - Knowing B's BD\_ADDR, A could attempt to exchange files with it using the OBEX (object exchange) protocol.
  - A session protocol that resembles HTTP, OBEX is used to transfer images, business cards, and other files between Bluetooth devices.
  - An attacker, A, could use OBEX Push to transfer a file containing malicious code to user, B.
  - User authorization is usually required before a file can be accepted by his smartphone. Each user selects a PIN which varies between 4 and 16 characters long but 4 characters are typically used.
  - The smartphone usually prompts a user to enter his/her PIN as a way to confirm whether an external file, for example, should be accepted.
  - Some OS versions accept file transfers without user authorization. And some smartphones allow users to disable the "Authorization Required" option for file transfers.

### *Link Level Security*

- The level of security provided by user authorization alone is generally inadequate.
- Another level of security provided by Bluetooth is link-level authentication and encryption.
- For this purpose, both sides compute a common secret called the link key.

- 
- 
- Bluetooth uses a procedure called **pairing** wherein this key is computed by two participating devices.
  - Pairing is preceded by discovery/inquiry and paging.
  - The latter is a procedure whereby the discovering device, A, establishes a connection with the discovered device, B.
  - Computing the Link Key
    1. The first step in deriving the common link key between A and B is to compute an initialization key,  $K_{init}$ .
      - ✓ This is a function of the  $BD\_ADDR$  of B and a nonce,  $IN\_RAND$ , generated by A as shown in Fig. 19.7(a).
      - ✓ Before the pairing procedure, the owners of A and B must agree in an off-line manner on a temporary PIN to be used specifically as part of the pairing procedure.
      - ✓ Both users then type in the temporary PIN.  $K_{init}$  is also a function of this temporary PIN agreed to by both parties.  $K_{init}$  is computed from  $IN\_RAND$ ,  $BD\_ADDR_0$ , and the PIN using an algorithm, E22, based on a block cipher called SAFER+.
    2. To compute the link key,
      - ✓ A and B each generate a random number ( $LK\_RAND_A$  and  $LK\_RAND_B$ , respectively).
      - ✓ Each party then performs an XOR of its random number with and they each transmit this across.
      - ✓ Each side recovers the other party's random number by performing an XOR of the received value with  $K_{init}$ , (see Fig. 19.7(6)).
      - ✓ Now, each side has  $LK\_RAND_A$ ,  $LK\_RAND_B$ , and the two device addresses,  $BD\_ADDR_A$  and  $BD\_ADDR_B$ .
      - ✓ They then perform identical operations on these to obtain the link key,  $K_{AB}$  as shown in Fig. 19.7(a). The operations involve the use of an algorithm, E21, which, like E22, is based on the cipher, SAFER+ .
      - ✓ Thereafter, each device stores the pair,  $BD\_ADDR$ , of the other device and the newly computed link key in a database. Each device maintains such a database of  $BD\_ADDR$ , link key pairs, one pair per device it is paired up with.

### *Using the Link Key*

- The link key is used for both authentication and encryption.
- Suppose A and B were already paired.
- Let  $K_A$ , be their common link key.
- Now suppose A wishes to authenticate B.

- For this purpose, it generates a random number,  $RAND_A$  (a challenge) and sends it to B.
- The response computed by B is  $E_1(K_{AB}, RAND_A, BD\_ADDR_B)$ .

### Hacking the link key

- It is possible to launch a dictionary attack by sniffing each message involved in pairing and authentication.
- These attacks enable an eavesdropper to obtain the link key  $K_{AB}$ .

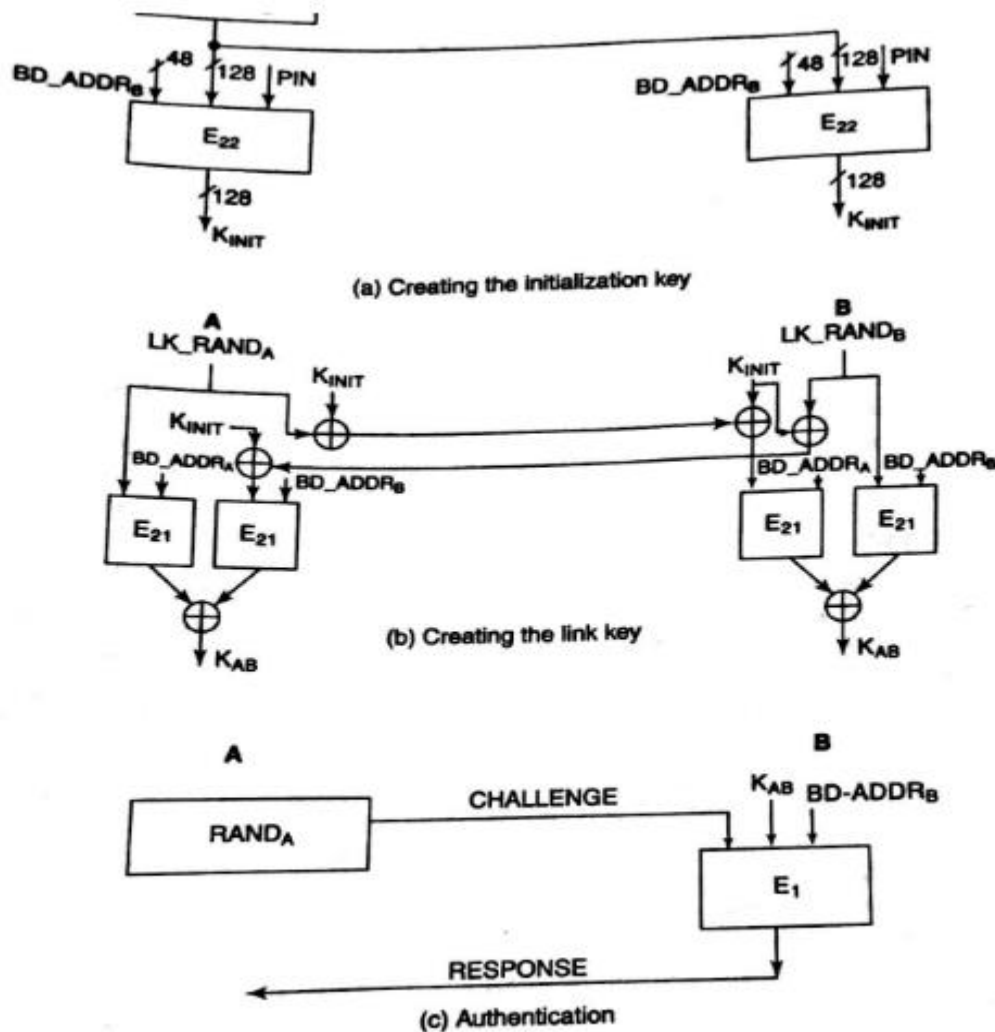


Figure 19.7 Generation and use of the Link key

### 19.6.3 Examples

- Cabir was one of the earliest proof-of concept worms that targeted the Symbian Series 60 OS.
- Unleashed in June 2004, it was authored by the International Virus writing group 29A.



- 
- The worm attempts to discover other Bluetooth-enabled phones set in discoverable mode.
  - When it finds such a phone, it sends the worm payload in a SIS file.
  - The receiver needs to accept and install the file.
  - Its payload was mostly benign typically displaying "Caribe" on the screen. However, the continuous scanning for new victims by an infected phone depletes battery power.
  - Commwarrior, which appeared in March 2005, was the first worm to spread through, both Bluetooth and MMS.
  - Like Cabir, it targeted Symbian smartphones.

## 19.7 BOTNETS

### 19.7.1 Basics

- A botnet is an army of compromised computers or bots connected to the Internet and remotely controlled by a "**botmaster.**"
- The earliest botnets were a collection of zombies that participated in DDoS attacks.
- The emergence of botnets is closely linked to the motive of financial gain that is behind many recent cyber attacks.
- They are often used to send spam mail on behalf of third parties,
- For example, Bot programs, may contain keyloggers and other forms of spyware that capture sensitive personal information such as passwords and credit card numbers and send these to the botmaster.
- Botnets have also been used as an extortion tool — "Pay up or your website will be bombarded by a DDoS attack".
- How does a computer become a bot?
- Bots are created in ways similar to many of the traditional trojan/worm/virus infections.
- A common vector of propagation is e-mail that contains an infected attachment.
- Another is through downloading a malicious webpage containing scripts that exploit vulnerabilities in certain browsers or application software.
- A bot infection may also be propagated by bots themselves by scanning the Internet for vulnerable machines.
- Finally, open file shares and IRC (Internet Relay Chat) multicast messages have also been widely used to spread infections.
- One important difference between a bot and a computer infected by a traditional worm/virus/ Trojan is that a bot needs to communicate with specific nodes in the botnet to receive fresh commands.
- A bot may be ordered to send spam or to "Launch a DDoS attack on site abc.com beginning 14:00 hours on 01-12-10." Some of the nodes in the botnet play the role of Command and Control (C&C) servers. They receive commands from the botmaster and disseminate these to the rest of the bots.

---

### 19.7.2 Case Study: The Storm Botnet

- The Storm botnet was first detected in January 2007.
- Its other names are Peacomm, Nuwar, and Zhelatin.
- Bots in the Storm botnet are infected in stages.
- The most common vectors for propagating the primary infection appear to be e-mail or infected websites. E-mail was sent with sensational subject lines like "230 die as Storm batters Europe."
- Likewise, users were lured into downloading free but infected files from websites containing music of various pop artists.
- The primary infection instructed the victim to join the Storm hornet embedded in the Overnet P2P network.
- Once part of the botnet, the bat was programmed to receive the second and subsequent injections of malicious code. One of the injections instructed the bat to propagate e-mail viruses. Another injection received some days later instructed the bat to launch a DDoS attacks on a target specified by the botmaster.

# WEB SECURITY

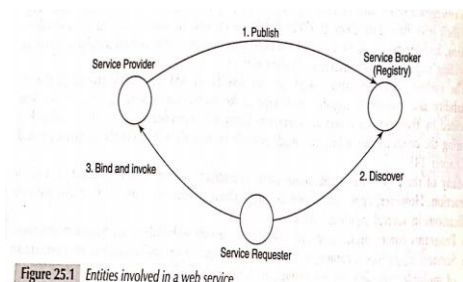
1

## Web services

- W3c defines a web service as:
- A software system identified by a URI whose public interfaces and bindings are defined and described by XML.
- The entities involved are:
- Providers register or publish their services in a public registry.
- Requesters discover services by querying the registry for services that match certain criteria.
- Once a requester has identified a provider whose services it needs ,it binds to and invokes the service of that provider.

2

## Entities involved in a web service



3

## XML

- XML is a meta markup language for text documents / textual data.
- XML allows to define languages („applications“) to represent text documents / textual data.
- Easy to understand for human users
- Very expressive (semantics along with the data)
- Well structured, easy to read and write from programs

April 29th, 2003

Organizing and Searching Information with XML

4

## Possible Advantages of Using XML

- Truly Portable Data
- Easily readable by human users
- Very expressive (semantics near data)
- Very flexible and customizable (no finite tag set)
- Easy to use from programs (libs available)
- Easy to convert into other representations (XML transformation languages)
- Many additional standards and tools
- Widely used and supported

5

## XML Documents

What's in an XML document?

- Elements
- Attributes
- plus some other details

6

## Elements in XML Documents

- (Freely definable) **tags**: `article`, `title`, `author`
  - with start tag: `<article>` etc.
  - and end tag: `</article>` etc.
- **Elements**: `<article> ... </article>`
- Elements have a **name** (`article`) and a **content** (`...`)
- Elements may be nested.
- Elements may be empty: `<this_is_empty/>`
- Each XML document has exactly one root element and forms a tree.

7

## XML by Example

```
<article>
  <author>Gerhard Weikum</author>
  <title>The Web in 10 Years</title>
</article>
```

8

## A Simple XML Document

```
<article>
  <author>Gerhard Weikum</author>
  <title>The Web in Ten Years</title>
  <text>
    <abstract>In order to evolve...</abstract>
    <section number="1" title="Introduction">
      The <index>Web</index> provides the universal...
    </section>
  </text>
</article>
```

9

## A Simple XML Document

```
<article>
  <author>Gerhard Weikum</author>
  <title>The Web in Ten Years</title>
  <text>
    <abstract>In order to evolve...</abstract>
    <section number="1" title="Introduction">
      The <index>Web</index> provides the universal...
    </section>
  </text>
</article>
```

Freely definable tags

10

## A Simple XML Document

```
<article>
  <author>Gerhard Weikum</author>
  <title>The Web in Ten Years</title>
  <text>
    <abstract>In order to evolve...</abstract>
    <section number="1" title="Introduction">
      The <index>Web</index> provides the universal...
    </section>
  </text>
</article>
```

Start Tag

End Tag

Element

Content of the Element (Subelements and/or Text)

11

## A Simple XML Document

```
<article>
  <author>Gerhard Weikum</author>
  <title>The Web in Ten Years</title>
  <text>
    <abstract>In order to evolve...</abstract>
    <section number="1" title="Introduction">
      The <index>Web</index> provides the universal...
    </section>
  </text>
</article>
```

Attributes with name and value

12

## Elements vs. Attributes

Elements may have **attributes** (in the start tag) that have a **name** and a **value**, e.g. `<section number="1">`.

What is the difference between elements and attributes?

- Only one attribute with a given name per element (but an arbitrary number of subelements)
- Attributes have no structure, simply strings (while elements can have subelements)

As a *rule of thumb*:

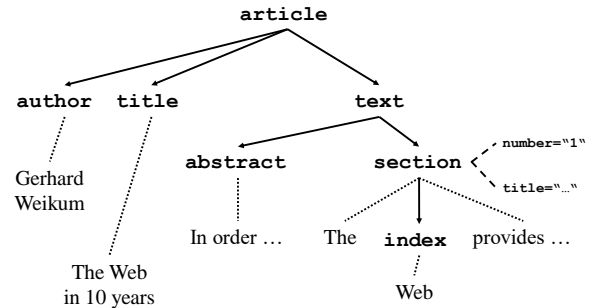
- Content into elements
- Metadata into attributes

Example:

```
<person born="1912-06-23" died="1954-06-07">
Alan Turing</person> proved that...
```

13

## XML Documents as Ordered Trees



14

## Well-Formed XML Documents

A **well-formed** document must adhere to, among others, the following rules:

- Every start tag has a matching end tag.
- Elements may nest, but must not overlap.
- There must be exactly one root element.
- Attribute values must be quoted.
- An element may not have two attributes with the same name.
- Comments and processing instructions may not appear inside tags.

15

## Namespace Syntax

```
<dbs:book xmlns:dbs="http://www-dbs/dbs">
```

Prefix as abbreviation  
of URI

Unique URI to identify  
the namespace

Signal that namespace  
definition happens

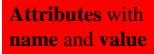
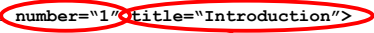
16

## 3.2 XML Schema Basics

- XML Schema is an XML application
- Provides simple types (string, integer, dateTime, duration, language, ...)
- Allows defining possible values for elements
- Allows defining types derived from existing types
- Allows defining complex types
- Allows posing constraints on the occurrence of elements
- Allows forcing uniqueness and foreign keys

## A Simple XML Document

```
<article>
  <author>Gerhard Weikum</author>
  <title>The Web in Ten Years</title>
  <text>
    <abstract>In order to evolve...</abstract>
    <section number="1" title="Introduction">
      The <index>Web</index> provides the universal...
    </section>
  </text>
</article>
```



18

## What is SOAP?

- SIMPLE OBJECT ACCESS PROTOCOL
- For exchanging structured information over internet.
- SOAP can be used over any transport protocol such as TCP, HTTP, SMTP
- SOAP defines a model for processing individual, one-way messages
- The mapping between soap message and an underlying transport protocol is referred as SOAP binding.
- Soap may run on top of http or SMTP.

19

## SOAP Message Format

- SOAP message consists of three parts:
    - SOAP Envelope
    - SOAP Header (optional)
    - SOAP Body
- SOAP Header:
- The Header element is a generic container for control information
  - Header blocks should contain information that influences payload processing
  - Header is optional
- SOAP Body:
- The Body element represents the message payload

20

```

<?xml version="1.0"?>
<soap:Envelope
xmlns:soap = "http://www.w3.org/2001/12/soap-envelope . . ." >
  <soap:Body xmlns:X="http://www.stockQuote.com/price">
    <X:GetPrice xmlns:X = "http://www. . ." >
      <X:StockName> MyStartUp </X:StockName>
    </X:GetPrice>
  </soap:Body>
</soap:Envelope>

```

(a) SOAP message in HTTP POST request

```

<?xml version="1.0"?>
<soap:Envelope
xmlns:soap = "http://www.w3.org/2001/12/soap-envelope . . ." >
  <soap:Body xmlns:X="http://www.stockQuote.com/price">
    <X:GetPriceResponse xmlns:X = "http://www. . ." >
      <X:Price> 3847 </X:Price>
    </X:GetPriceResponse>
  </soap:Body>
</soap:Envelope>

```

(b) SOAP message in HTTP response

## WSDL

- Web services Definition language
- “An XML format for describing network services as a set
- Contains information where the service is located, what the service does, and how to invoke the service such as types ,messages,operation,port types,and bindings.
- Operation:abstract definition of a n action.
- Message: abstract definition of data beingexchanged as a part of operation.

## WSDL

```

<message name="message1">
  <part name=" . . ." type=" . . ." />
</message>
<message name="message2">
  <part name=" . . ." type=" . . ." />
</message>
<portType name="portType1">
  <operation name=" . . ." >
    <input message="message1"/>
    <output message="message2"/>
  </operation>
</portType>

```

Port type that includes one operation comprising two messages



## UDDI

- UNIVERSAL DESCRIPTION DISCOVERY AND INTEGRATION
- IS a registry or catalogue that allows businesses across the globe to list themselves in internet.
- Clients query the registry services
- Response :access to WSDL.

25

## WS security

- Token Types
- XML Encryption
- XML signature

26

## Token types

- Web security addresses basic problems in securing messages used in web services.
- Its main functions are:
  - It defines XML elements that are used to communicate *security tokens* (defined below) in the header of a SOAP message.
  - Together with the XML Encryption Standard it defines the syntax and processing rules used to *encrypt* one or more parts of a SOAP message.
  - Together with the XML Signature Standard, it defines the syntax and processing rules used to create and represent a *digital signature* on one or more parts of a SOAP message.

27

## Token types

The user name, hash, nonce, and timestamp are sent in a <UsernameToken>

```
SHA-1 (n, t, pw)
< UsernameToken >
  < Username > John </Username >
  < Password Type = "PasswordDigest" >
    4u%h@+q:L
  </Password >
  < Nonce > . . . </Nonce >
  < Created > . . . </Created >
</UsernameToken>
```

28

## Token types

```
< Security >  
  ...  
  < BinarySecurityToken  
    ValueType = " ... X509v3"  
    EncodingType = " ... Base64Binary" >  
    Lp9tba4Pc7G ...  
  < / BinarySecurityToken >  
  ...  
< / Security >
```

29

## XML Encryption

- It defines XML elements for representing encrypted data and keys used for encryption.
- Encryption at different levels of granularity
- An entire document
- A complete XML element within a document
- Content of an XML element.

30

## XML Encryption

- <EncryptedData> element contains:
  - A Type attribute – indicates the type of the information encrypted
  - Information about the algorithm used for encryption
  - <CipherData> A Reference to the cipher, or the cipher itself
  - <EncryptedKey> - used for encrypting pre shared key or session key.

31

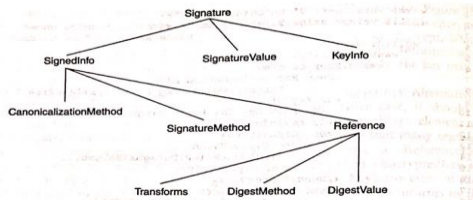
## XML Encryption

```
< S11:Body wsu:Id = "#Id1" >  
  <xenc:EncryptedData  
    Type = "http://www.w3.org/2001/04/xmlenc#Element"  
    Id="#Id2">  
    <xenc:EncryptionMethod  
      Algorithm = "http://www.w3.org/xmlenc#aes256-cbc"/>  
    <xenc:CipherData>  
      <xenc:CipherValue>  
        tdaqUsjXipJ09jlkjh5oinlkdsn...  
      </xenc:CipherValue>  
    </xenc:CipherData>  
  </xenc:EncryptedData>
```

32

## XML signature

- Specifies syntax for signatures and signature keys.
- Canonical form: transformed to
- Xml Signature is included in header of soap message.



5.6 Tree for signature element

33

## XML signature

- Signed Info

<SignedInfo>

Within this element is included information about the *canonicalization algorithm* and *signature algorithm* employed. An example of the signature algorithm is RSA-SHA1, i.e., the use of SHA-1 to perform the hash followed by an RSA private key operation on the hash value. A "signature" in the form of a Message Authentication Code (MAC) is also supported.

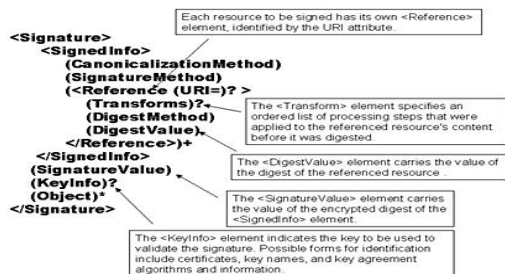
- Signature Value

- Contains digital signature value.
- On Entire signature element

- Key Info

- the key required to verify digital signature at the receiver end.

34



35

```

<Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
  <SignedInfo>
    <CanonicalizationMethod
      Algorithm="http://www.w3.org/2001/10/xmlsec14n#" />
    <SignatureMethod
      Algorithm="http://www.w3.org/2000/xmldsig#rsa-sha1" />
    <Reference URI="#Id1">
      <Transforms>
        <TransformAlgorithm =
          "http://www.w3.org/2001/xmlsec14n#" />
      </Transforms>
      <DigestMethodAlgorithm=
        "http://www.w3.org/2000/09/xmldsig#sha1"/>
      <DigestValue> Yhs1 . . . pKl </DigestValue>
    </Reference>
  </SignedInfo>
  <SignatureValue>
  <KeyInfo>
  </Signature>
  
```

36

### SAML:security assertions markup language

- SAML is the XML based standard created to enable portable identities and the assertions these identities want to make.
- Defines a standard message exchange protocol. Specifies how it is transported.
- Single Sign-On (SSO)
- Portable Trust” - a user, whose identity is established and verified in one domain, can invoke services in another domain

37

### SAML Assertions

- SAML defines three types of assertions.
  1. Authentication
  2. Authorization
  3. Attributes

38

### SAML

- Assertion is a claim, statement, or declaration of fact made by SAML authority
- Types of assertions:
- Authentication - the subject was authenticated by a particular means at a particular time
- Authorization - the subject was granted or denied access to a specified resource
- Attributes -the subject is associated with the supplied attribute.

39

### Common Elements

- **<Issuer>** - the issuer name **[Required]**
- **<ds:Signature>** - an XML signature for integrity protection and authentication of the issuer **[Optional]**
- **<Subject>** - the subject of the statements in the assertion **[Optional]**
- **<Conditions>** - must be evaluated when using assertions **[Optional]**

40

## SAML code

```

1 <saml:Assertion xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion" Version = "2.0"
2   IssuanceInstant = "2010-02-01T08:25:15Z">
3   <saml:Issuer Format="urn:oasis:names:tc:SAML:2.0:uri" URI="http://www.admin.iitb.ac.in"/>
4   <saml:Subject>
5     <saml:NameID Format="urn:oasis:names:tc:SAML:2.0:emailAddress" Value="rajeshX@cse.iitb.ac.in"/>
6     </saml:NameID>
7   </saml:Subject>
8   <saml:Conditions>
9     <saml:NotBefore NotBefore="2010-02-01T08:26:00Z" NotOnOrAfter="2010-02-02T10:30:00Z"/>
10    </saml:NotBefore>
11  </saml:Conditions>
12  <saml:AuthnStatement AuthnInstant="2010-02-01T08:25:15Z" SessionIndex="1234">
13    <saml:AuthnContext>
14      <saml:AuthnContextClassRef URI="urn:oasis:names:tc:SAML:2.0:authn-PasswdProtectedTransport"/>
15      </saml:AuthnContextClassRef>
16    </saml:AuthnContext>
17  </saml:AuthnStatement>
18 </saml:Assertion>

```

41

## Example of SAML

Now suppose that Sandeep is a gold customer of SmartTravels – a status conferred on all customers of SmartTravels who have done business in excess of Rs. 4,00,000 over the last 4 years. SmartTravels has business relationships with several airlines, including Jet Air, which provide varying discounts to all of its gold customers. How is Jet Air expected to know that Sandeep is a gold customer of SmartTravels and is eligible for the discounted price?

For the sake of completeness, we enumerate all the steps involved in Sandeep's transaction.

1. Sandeep logs in to the SmartTravels website and is *authenticated*. He indicates the destination city, date and time of travel, and price of ticket he is willing to pay.
2. SmartTravels determines that Sandeep is a gold customer and presents a list of airlines that satisfy Sandeep's requirements.
3. Sandeep clicks on the airline of interest, say JetAir.
4. SmartTravels creates *SAML assertions* indicating that
  - a. Sandeep has been authenticated using a login name–password mechanism (authentication assertion)
  - b. Sandeep is a gold customer (attribute assertion)
5. SmartTravels creates an HTML form with two hidden inputs. The first, named *SAMLResponse*, contains the signed SAML assertion. The second hidden input, called *RelayState*, contains the *URL of the resource* required by Sandeep. The relevant portion of the form is

42

## Other standards

- WS trust
- WS security policy

43

## WS trust

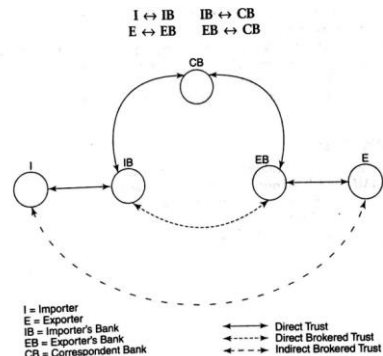


Figure 25.9 Trust relationship between entities involved in international trade

44

### WS trust could be used in the following manner:

- I authenticates himself to IB.
- Since IB and CB trust each other, IB requests CB to issue a security token to be used by I.
- CB creates a security token and includes information such as the maximum credit amount extended to I. CB acts as a Security Token Service Provider. CB communicates this token to IB who forwards it to I.
- I dispatches the token to E.
- E requests EB to validate the token. EB may be able to validate the token on its own. If not, it sends it to CB for validation.
- The success or failure of the validation process is communicated by CB to E through EB.

45

### Web security policy

- Web security policy enables a web service to specify the security tokens it will accept for authentication and access control.
- Returning to the example of Fig. 25.9, the policies regarding authentication laid out by relevant entities may be as follows:
- An importer must authenticate himself to his local bank via a login name and password.
- A local bank must authenticate itself to the global bank using a challenge-response protocol in conjunction with a digital certificate.
- An importer must authenticate himself to the given exporter through a SAML token signed by a global bank.

46