B. TEACH (SEVENTH SEMESTER)

COMPUTER SCIENCE & ENGG. BRANCH

Note : Attempt question of all two sections as directed.

SECTION A

Attempt all question. Each question carries 2 mark

Choose the correct answer

1: In a(n) ________, the key is called the secret key.

a. symmetric-key
b. asymmetric-key
c. either (a) or (b)
d. neither (a) nor (b)

The correct answer is a

2: In an asymmetric-key cipher, the receiver uses the ______ key.

a. private
b. public
c. either (a) or (b)
d. neither (a) nor (b)

The correct answer is a

3: A ________ cipher replaces one character with another character.

a. substitution
b. transposition
c. either (a) or (b)
d. neither (a) nor (b)

The correct answer is a

4: _________ ciphers can be categorized into two broad categories: monoalphabetic and polyalphabetic.

a. Substitution
b. Transposition
c. either (a) or (b)
d. neither (a) nor (b)

The correct answer is a

5: Message_____ means that the data must arrive at the receiver exactly as sent.

a. confidentiality

b. integrity
c. authentication
d. none of the above

The correct answer is b

6: ________ means to prove the identity of the entity that tries to access the system's resources.

a. Message authentication
b. Entity authentication
c. Message confidentiality
d. none of the above

The correct answer is b

7: A(n) _________ can be used to preserve the integrity of a document or a message.

a. message digest
b. message summary
c. encrypted message
d. none of the above

The correct answer is a

8: _________ is a collection of protocols designed by the IETF (Internet Engineering Task Force) to provide security for a packet at the network level.

a. IPSec
b. SSL
c. PGP
d. none of the above

The correct answer is a

9: The _______ mode is normally used when we need host-to-host (end-to-end) protection of data.

a. transport
b. tunnel
c. either (a) or (b)
d. neither (a) nor (b)

The correct answer is a

10: In the _______ mode, IPSec protects the whole IP packet, including the original IP header.

a. transport
b. tunnel
c. either (a) or (b)
d. neither (a) nor (b)

The correct answer is b

UNIT I

Ans 2 We need an even number of letters, so append a "q" to the end of the message. Then convert the letters into the corresponding alphabetic positions:

m	e	e	t	m	e	a	t	t	h	e	u	s	u	a	l
13	5	5	20	13	5	1	20	20	8	5	21	19	21	1	12
p	l	a	c	e	a	t	t	e	n	r	a	t	h	e	r
16	12	1	3	5	1	20	20	5	14	18	1	20	8	5	18
t	h	a	n	e	i	g	h	t	o	c	l	o	c	k	q
20	8	1	14	5	9	7	8	20	15	3	12	15	3	11	17

The calculations proceed two letters at a time.

The first pair:

$C1C2 = 9457 \quad 135 \quad \mathrm{mod}26 = 137100 \quad \mathrm{mod}26 = 722$

The first two ciphertext characters are alphabetic positions 7 and 22, which correspond to GV. The complete ciphertext:

GVUIGVKODZYPUHEKJHUZWFZFWSJSDZMUDZMYCJQMFWWUQRKR

Or

Ans 3

Model for Network Security

A message is to be transferred from one party to another across some sort of Internet service.

A logical information channel is established by defining a route through the Internet from source to destination and by the cooperative use of communication protocols. Security aspects come into play when it is necessary or desirable to protect the information transmission from an opponent who may present a threat to confidentiality, authenticity, and so on.

All the techniques for providing security have two components:

 A security-related transformation on the information to be sent.

Examples include the encryption of the message, which scrambles the message so that it is unreadable by the opponent, and the addition of a code based on the contents of the message, which can be used to verify the identity of the sender.

 Some secret information shared by the two principals and, it is hoped, unknown to the opponent.

An example is an encryption key used in conjunction with the transformation to scramble the message before transmission and unscramble it on reception.

 A trusted third party may be needed to achieve secure transmission.

For example, a third party may be responsible for distributing the secret information to the two principals while keeping it from any opponent. Or a third party may be needed to arbitrate disputes between the two principals concerning the authenticity of a message transmission.

This general model shows that there are four basic tasks in designing a particular security service:
1. Design an algorithm for performing the security-related transformation. The algorithm should be such that an opponent cannot defeat its purpose.

2. Generate the secret information to be used with the algorithm.

3. Develop methods for the distribution and sharing of the secret information.

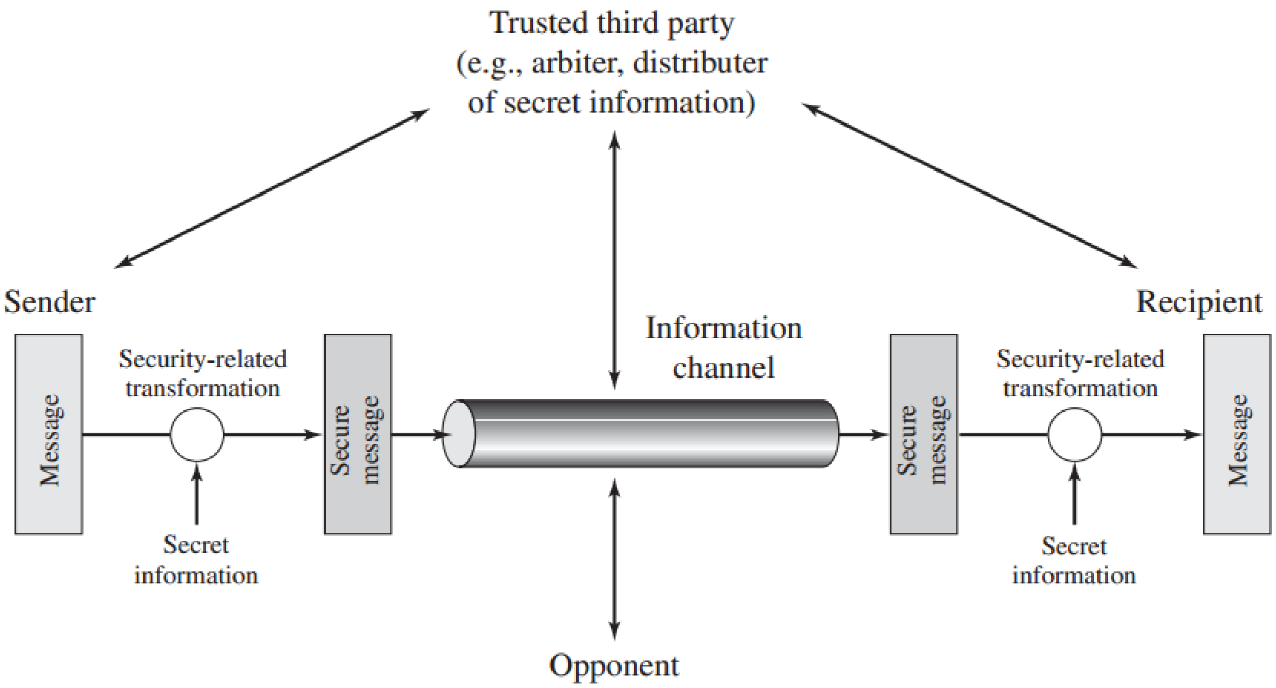

Figure: A Model for Network Security

4. Specify a protocol to be used by the two principals that makes use of the security algorithm and the secret information to achieve a particular security service.

Monoalphabetic Ciphers
Here is an example of a plaintext – ciphertext alphabet pair for each type of cipher we have seen thus far.
1. A Caesar cipher with key 5:
plaintext alphabet
abcdefghijklmnopqrstuvwxyz
ciphertext alphabet
FGHIJKLMNOPQRSTUVWXYZABCDE
2. A Decimation Cipher modulo 26 with key 21:
plaintext alphabet
abcdefghijklmnopqrstuvwxyz
ciphertext alphabet

UPKFAVQLGBWRMHCXSNIDYTOJEZ
3. A Linear Cipher modulo 26 with key 7
m+ 9:
plaintext alphabet
abcdefghijklmnopqrstuvwxyz
ciphertext alphabet
PWDKRYFMTAHOVCJQXELSZGNUBI
These are all monoalphabetic ciphers, ciphers in which the same plaintext letters are always replaced by the same ciphertext letters.Mono, meaning one, indicates that each letter has a single substitute. In this chapter we look at other ways of creating monoalphabetic ciphers.
To construct a monoalphabetic cipher, we need to create some ordering of the alphabet, such as SOMERDINGXHBVLTUJWKYZFACPQ, and pair it with a plaintext alphabet,
plaintext alphabet
abcdefghijklmnopqrstuvwxyz
ciphertext alphabet
SOMERDINGXHBVLTUJWKYZFACPQ
.
However it is not particularly easy to remember apparently random orderings of 26 letters. So we will concentrate a couple of well-known methods that use akey to develop the ciphertext alphabet's order

UNIT II

Ans 4

the Triple-DES algorithm, in which DES is applied three times. If we consider a triple length key to consist of three 56-bit keys K1, K2, K3 then encryption is as follows:

- Encrypt with K1
- Decrypt with K2
- Encrypt with K3

Decryption is the reverse process:

- Decrypt with K3
- Encrypt with K2
- Decrypt with K1

Setting K3 equal to K1 in these processes gives us a double length key K1, K2.

Setting K1, K2 and K3 all equal to K has the same effect as using a single-length (56-bit key). Thus it is possible for a system using triple-DES to be compatible with a system using single-DES.

when you compose a cipher into a new one, you can't use a double enciphering. There is a class of attacks called meet-in-the-middle attacks, in which you encrypt from one end, decrypt from the other, and start looking for collisions (things that give you the same answer). With sufficient memory, Double DES (or any other cipher) would only be twice as strong as the base cipher -- or one bit more in strength.

remember that the reason we're going through this multiple-encryption exercise is because we want to make a composite cipher that is stronger than single DES. Because of the meet-in-the-middle attack, double DES is only one-bit stronger than single DES. Two-key triple DES thus has 112 bits of strength. But what about the three-key version of triple DES? Common sense dictates it would be at least as strong as two-key triple DES, but how much stronger?

The answer is that no one knows. I've seen arguments suggest Triple DES always has 112 bits of strength. I've seen them that it has the full 168 bits. (Note that we're ignoring the obvious weak keys, like K1=K2.) I don't like either, myself, and actually think that the ones that you don't ever get more than 112 bits are better arguments, even though I disagree.

One thing to remember is that in cryptography there's a difference between a theoretic attack and a real one. Let's suppose, for example, I came up with an attack that needed 2^80 cipher blocks, and then could always make three-key Triple DES be no stronger than 112 bits.

or

Ans 5

Blowfish Encryption Algorithm

Blowfish is a symmetric block encryption algorithm designed in consideration with,

- **Fast :** It encrypts data on large 32-bit microprocessors at a rate of 26 clock cycles per byte.

- **Compact:** It can run in less than 5K of memory.

- **Simple:** It uses addition, XOR, lookup table with 32-bit operands.

- **Secure:** The key length is variable ,it can be in the range of 32~448 bits: default 128 bits key length.

- It is suitable for applications where the key does not change often, like communication link or an automatic file encryptor.

- Unpatented and royality-free.

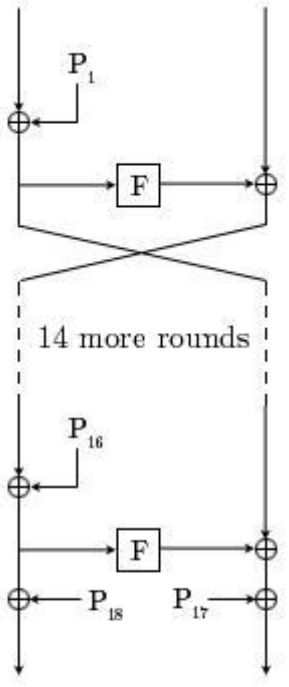

Fig 1: The Feistel structure of Blowfish

[Source: http://en.wikipedia.org/wiki/File:BlowfishDiagram.png]

2.1 Description of Algorithm:

Blowfish symmetric block cipher algorithm encrypts block data of 64-bits at a time.it will follows the feistel network and this algorithm is divided into two parts.

1. Key-expansion

2. Data Encryption

2.1.1 Key-expansion:

It will converts a key of at most 448 bits into several subkey arrays totaling 4168 bytes. Blowfish uses large number of subkeys.

These keys are generate earlier to any data encryption or decryption.

The p-array consists of 18, 32-bit subkeys:

P1,P2,…………,P18

Four 32-bit S-Boxes consists of 256 entries each:

S1,0, S1,1,………. S1,255

S2,0, S2,1,……….. S2,255

S3,0, S3,1,……….. S3,255

S4,0, S4,1,..............S4,255

Generating the Subkeys:

The subkeys are calculated using the Blowfish algorithm:

1. Initialize first the P-array and then the four S-boxes, in order, with a fixed string. This string consists of the hexadecimal digits of pi (less the initial 3): P1 = 0x243f6a88, P2 = 0x85a308d3, P3 = 0x13198a2e, P4 = 0x03707344, etc.
2. XOR P1 with the first 32 bits of the key, XOR P2 with the second 32-bits of the key, and so on for all bits of the key (possibly up to P14). Repeatedly cycle through the key bits until the entire P-array has been XORed with key bits. (For every short key, there is at least one equivalent longer key; for example, if A is a 64-bit key, then AA, AAA, etc., are equivalent keys.)
3. Encrypt the all-zero string with the Blowfish algorithm, using the subkeys described in steps (1) and (2).
4. Replace P1 and P2 with the output of step (3).

5. Encrypt the output of step (3) using the Blowfish algorithm with the modified subkeys.
6. Replace P3 and P4 with the output of step (5).
7. Continue the process, replacing all entries of the P array, and then all four S-boxes in order, with the output of the continuously changing Blowfish algorithm.

In total, 521 iterations are required to generate all required subkeys. Applications can store the subkeys rather than execute this derivation process multiple times.

2.1.2 Data Encryption:

It is having a function to iterate 16 times of network. Each round consists of key-dependent permutation and a key and data-dependent substitution. All operations are XORs and additions on 32-bit words. The only additional operations are four indexed array data lookup tables for each round.

```
-------------------------------------------------

 Algorithm:Blowfish Encryption

   --------------------------------------------------------------

Divide x into two 32-bit halves: xL, xR

          For i = 1to 16:

                 xL = XL XOR Pi

                 xR = F(XL) XOR xR

                 Swap XL and xR

                 Swap XL and xR (Undo the last swap.)
```

```
        xR = xR XOR P17

    xL = xL XOR P18

    Recombine xL and xR
```

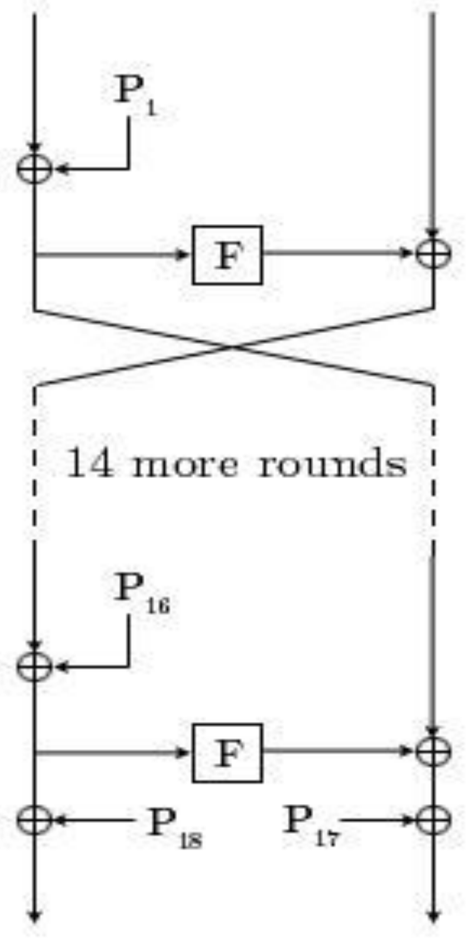

Fig 2: Blowfish Encryption

Addition: Addition of words, denoted by +, is performed modulo 2*w*. The inverse operation, denoted by –, is subtraction modulo 2*w*.

Bitwise exclusive-OR: This operation is denoted by "⊕".

Left circular rotation: The cyclic rotation of word *x* left by *y* bits is denoted by *x* <<< *y*. The inverse is the right circular rotation of word *x* by *y* bits, denoted by *x* >>> *y*.

UNIT III

Encryption/decryption: The sender encrypts a message with the recipient's public key.

Digital signature: The sender "signs" a message with its private key. Signing is achieved by a cryptographic algorithm applied to the message or to a small block of data that is a function of the message.

Key exchange: Two sides cooperate to exchange a session key. Several different approaches are possible, involving the private key(s) of one or both parties.

RSA Algorithm

The RSA Algorithm was first published in the paper named "*A method for obtaining Digital Signatures and Public-Key Cryptosystems*" in 1977 by Ron Rivest, Adi Shamir and Leonard Adleman.
RSA uses two exponents, e and d, where 'e' is public and 'd' is private. Suppose P is the plaintext block and C is the ciphertext block.
Alice uses C = Pe mod n to create ciphertext C from plaintext P. Bob uses P = Cd mod n to retrieve the plaintext sent by Alice.
Both the sender and the receiver must know the value of n. Thus, this is a public-key encryption algorithm with a public-key of PU = {e,n} and a private-key of PR = {d,n}.
Key Generation Algorithm
1. Choose two very large random prime integers: p and q.

2. Compute n and Ø(n):

n = p x q and Ø(n) = (p-1) x (q-1).
[Ø(n) is called the *Euler Totient function*]
3. Choose an integer e, 1<e< Ø(n) such that gcd(e, Ø(n)) = 1.

4. Compute d, 1<d< Ø(n) such that e x d = 1 (mod Ø(n)).

Where,
□□The public key is (n,e) and the private key is (n,d).
□□The values of p, q and Ø(n) are private.
□□e is the public or encryption component.
□□d is the private or decryption component.

Example

1. Select the prime integers: p = 11, q = 3.

2. N = p x q = 33.

Ø(n) = (p-1) x (q-1) = 20.

3. Choose e = 3.

Check gcd(3,20) = 1.

4. Compute d = 7.

3 x d = 1 mod 20

→ d = 7.

The public key = (33, 3)

The private key = (33, 7) **Plaintext**	**Numeric**	**P3**	**C = P3 mod 33**	**C7**	**P = C7 mod 33**
S	19	6859	28	13492928512	19
U	21	9261	21	1801088541	21
Z	26	17576	20	12800000	26
A	01	1	1	1	01
N	14	2744	5	78125	14
N	14	2744	5	78125	14
E	05	125	26	8031810176	05

Or

Ans

Public-Key Distribution of Secret Keys

- use previous methods to obtain public-key
- can use for secrecy or authentication
- but public-key algorithms are slow
- so usually want to use private-key encryption to protect message contents
- hence need a session key
- have several alternatives for negotiating a suitable session

1. The distribution of public keys.
2. The use of public-key encryption to distribute secret keys

UNIT IV

A session state is defined by the following parameters (definitions from the SSL specification):
• Session identifier:
An arbitrary byte sequence chosen by the server to identify an active or resumable session state.
• Peer certificate:
An X509.v3 certificate of the peer. This element of the state may be null.
• Compression method:
The algorithm used to compress data prior to encryption.
• Cipher spec:
Specifies the bulk data encryption algorithm (such as null, DES, etc.) and a hash algorithm (such as MD5 or SHA-1) used for MAC calculation. It also defines cryptographic attributes such as the hash_size.
• Master secret:
Forty-eight-byte secret shared between the client and server.
• Is resumable:
A flag indicating whether the session can be used to initiate new connections.
A connection state is defined by the following parameters:
• Server and client random:
Byte sequences that are chosen by the server and client for each connection.
• Server write MAC secret:
The secret key used in MAC operations on data sent by the server.
• Client write MAC secret:
The secret key used in MAC operations on data sent by the client.
• Server write key:
The conventional encryption key for data encrypted by the server and decrypted by the client.

• Client write key:
The conventional encryption key for data encrypted by the client and decrypted by the server.
• Initialization vectors:
When a block cipher in CBC mode is used, an initialization vector (IV) is maintained for each key. This field is first initialized by the SSL Handshake Protocol. Thereafter the final ciphertext block from each record is preserved for use as the IV with the following record.
• Sequence numbers:
Each party maintains separate sequence numbers for transmitted and received messages for each connection. When a party sends or receives a change cipher spec message, the appropriate sequence number is set to zero. Sequence numbers may not exceed

---

Or

SSL Record Protocol

Different Layers in SSL, has got different responsibilities to full fill. SSL Record Layer protocol has got the below functions to fulfill.

- Breaking Down the Data from Application layers, with fixed length.
- Compress the Data
- Add Message Authentication Code, Which is calculated with the help of Integrity Key.
- Encrypt the packets(which was broked down with fixed length).
- Add SSL header's in the packets with fixed length. Which consists the following headers, which combinely form a 5byte header.

Record Protocol Header contents in SSL

1. 1 Byte Protocol Definition
2. 2 Byte Protocol version
3. 2 Byte Length

You can say that SSL Record Layer Protocol comes just above, the TCP or Transport Layer in TCP/IP protocol Stack, Which is evident from the below picture.

Record Layer in SSL is the layer, which provides the facilities like confidentiality through encryption and integrity check using MAC.

The object created by the record layer, by fragmenting the data from application layer, and adding appropriate headers, is called as a **record**.

MAC information is very much necessary in this layer, as it checks the integrity of the record. This is nothing but, a hash value calculated with the help of the following things.

MAC value is the Hash value calculated from [sequence number, padding, primary data, secret key].

So now our record will consist of the following things.

1. Data Fragment
2. Some Padding
3. MAC value

Confidentiality: The Handshake Protocol defines a shared secret key that is used for conventional encryption of SSL payloads.

Message Integrity: The Handshake Protocol also defines a shared secret key that is used to form a message authentication code (MAC).

UNIT V

Statistical anomaly detection involves the collection of data relating to the behavior of legitimate users over a period of time. Then statistical tests are applied to observed behavior to determine with a high level of confidence whether that behavior is not legitimate user behavior.

Rule-Based Detection involves an attempt to define a set of rules that can be used to decide that a given behavior is that of an intruder.

Statistical Anomaly Detection

Statistical anomaly detection techniques fall into two categories:

threshold detection and profile-based systems.

Threshold detection involves counting the number of occurrences of a specific event type over an interval time.

If the count surpasses what is considered a reasonable number that one might expect to occur, then the intrusion is assumed.

The threshold detection is not efficient.

Profile-based anomaly detection focuses on characterizing the past behavior of individual users or related groups of users and then detecting significant deviations.

A profile may consist of a set of parameters , so that deviation on just a single parameter may not be sufficient in itself to signal an alert.

The foundation of this approach is an analysis of audit records.

To do that, the designer decides on a number of quantitative metrics that can be used to measure user behavior.

Then current audit records are input to detect intrusion

Examples of metrics for profile-based intrusion detection:

•

Counter: times of logins, command executed during a single user session, number of password failures, etc.

•

Gauge: the number of logical connections assigned to a user application, the number of outgoing messages queued for a user process, etc.

•

Interval timer: the length of time between successive logins to an account, etc.

•

Resource utilization: number of pages printed during a user session, total time consumed by a program execution, etc.Using the metrics, various tests can be performed to determine whether current activities fit within acceptable limits.

•

Mean and standard deviation.

•

Multivariate

•

Markov process

•

Time series

•

Operational

Rule-Based Intrusion Detection

Rule-based techniques detect intrusion by observing events in the system and applying a set of rules that lead to a decision regarding whether a given pattern of activity is suspicious.

Rule-based anomaly detection: historical audit records are analyzed to identify usage patterns and to generate automatically rules that describe those patterns.

Similar to statistical anomaly detection, this method does not require knowledge of security vulnerabilities within the system. A rather large database of rules will be needed.

Rule-based penetration identification: use rules for identifying known penetrations or penetrations that would exploit known weaknesses. Rules can also be defined that identify suspicious behaviors.

Examples:

Users should not read files in other users' personal directories. Users must not write other users' files. Users who log in after hours often access the same files they used earlier.Users should not be logged in more than once to the same system.Users do not make copies of system programs.

Or

A **traditional packet filter** makes filtering decisions on an individual packet basis and does not take into consideration any higher layer context.

A **stateful inspection packet filter** tightens up the rules for TCP traffic by creating a directory of outbound TCP connections,.

There is an entry for each currently established connection. The packet filter will now allow incoming traffic to high-numbered ports only for those packets that fit the profile of one of the entries in this directory

In general, firewalls that make use of stateful inspection are the industry norm. Stateful inspection replaced packet filtering in most environments several years ago, and the majority of modern firewall systems take advantage of it.

The main difference between the two firewalls is that stateful inspection systems maintain a state table, allowing them to keep track of all open connections through a firewall, while packet-filtering firewalls do not. When traffic arrives, the system compares the traffic to the state table, determining whether it is part of an established connection.

The only place you'll likely see packet filtering in today's environment is at an Internet-facing router. These devices often implement a basic packet-filtering rule set to weed out obviously unwanted traffic and reduce the load on a stateful inspection firewall immediately behind the router.