Chapter 6

Thursday, 29 March 2018 1:34 AM

Chapter 7

Wednesday, 28 March 2018 10:56 PM

TCP/IP Stack
The Internet and Intranet are built using a group of protocols called the TCP/IP stack.

The flaws with TCP/IP are:
1) Reliance on IP source address for authentication - typical network communications like email and file transfer need the source and destination IP. The source IP can be spoofed and the intruder could access the system and services

2) Lack of authentication strategies for management protocols - Such are flow control, congestion control and routing protocols. Without proper authentication, attackers can use the protocols to configure and control the network.

3) Lack of confidentiality - The TCP/IP stack makes no effort to encrypt the data so someone using a sniffer like WireShark can see the packet.

To fix this there were 2 approaches:
1) Putting protocols before the Transport layer which enhanced TCP communication by adding things to enforce confidentiality, data integrity and authentication. The protocols are Transport Layer Security (TLS) and Secure Socket Layer (SSL).

2) Putting protocols before the Network layer such as the Internet Security (IPSec) protocol.

SSL Change Cipher Spec Protocol
After computation of session keys for given SSL session the CCS Protocol is responsible for periodically renegotiation new session info.

SSL Alert Protocol
This is used to report errors such as unexpected messages, bad record MAC, security parameters, negotiation failure and other type of errors.

SSL/TSL Threats
The most common attacks are SSL Stripping, STARTTLS Command Injection Attack, Padding Oracle Attacks and Browser Exploit Against SSL/TLS Attack

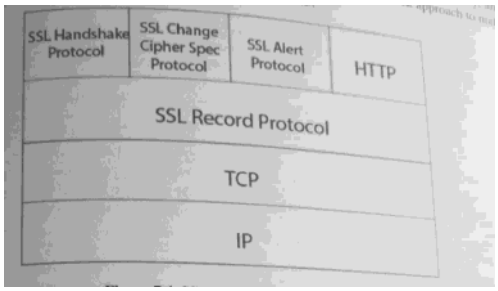
IPSec
IPSec provides security in three situations: host-to-host, host-to-gateway and gateway-to-gateway.

In host-to-host connection, two devices use the network and create a secure tunnel to each other.
In host-to-gateway scenario, host device has a secure connection to gateway.
In gateway-to-gateway, it is assumed that the access networks are trusted and only a secure channel is setup between gateways involved

Internet Key Exchange is responsible for mutual authentication and creation of the session key

Encapsulated security payload (ESP) and Authentication header (AH) provide different types of services. ESP provides encryption, authentication and integrity protection for IP packets. AH only provides authentication and integrity.

Transport Approach
SSL Handshake Protocol



- The SSL Handshake Protocol uses public key cryptography to establish session keys between the client and server.
- It is the first protocol executed between communicating parties
- It is responsible for:
  - Authentication process between client and server
  - Negotiation for encryption of MAC algorithms
  - Creation of session keys to be used to secure communication between client and server

Steps are:
1) After normal TCP 3-way handshake, client sends a clientHello message containing a randomly generated 32 bit nounce (Nc), protocol version, session ID, list of ciphers and list of compression methods

2) If clientHello is accepted, the server sends a serverHello message containing server certificate, a randomly generated 32 bit nonuce (Ns), protocol version, session ID, selected cipher and method of compression. Can also ask for client certificate in response message.

3) Client generates secret key material and sends it to server encrypted with public key.
   Key material is made up of:
   a. A pre-master secret S selected by client
   b. Calculated key K, K = h(S, Nc, Ns)

4) Server decrypts received info using its private key and obtains secret key material. Now both client and server share same secret key material

5) Generation of number of keys used for different security aspects of the SSL protocol. Both client an dserver use secret key material to generate 4 keys:
   a. Ec: Client -> Server data encryption key
   b. Es: Server -> Client data encryption key
   c. Mc: Client -> Server MAC key
   d. Ms: Client -> Client MAC key

6) Client sends change_cipher_spec
7) Client sends finished message
8) Server sense change_cipher_spec
9) Server sends finished message

Assignment 2 Notes

Thursday, 5 April 2018 1:47 AM

Q2
Since the page replacement is used with four page frames then there are 4 rows and an extra for page faults.
The FIFO approach is that when a page that hasn't been loaded into memory comes up, the first page that was loaded will be replaced
The FIFO approach is very simple cause it only requires a pointer to hold the location of the next page to be replaced

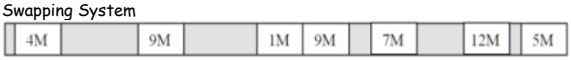
The Least Recently Used (LRU) replaces the page in memory that has not been referenced in the longest time. By principle of locality, this page should be the least likely to be referenced. An approach to this is to stamp the page with the time it was last refenced. This causes a lot of over head

Although LRU is realizable, few machines have the hardware to do it. One solution can be implemented in the software which is Not Frequently Used(NFU).
A software counter, initially 0 is associated with each page. Each page also has an R (reference) bit which is 0 if the page isn't being referenced and 1 if it is. At each clock interrupt, the OS scans all the pages in memory and adds the R bit to the counter. The counters keep track of how often each page has been referenced. When a page fault occurs, the page with the lowest counter is chosen for replacement.
The problem is that it never forgets, so pages that were heavily used during pass 1 may still have a high count into later passes. Consequentially, the OS will remove useful pages instead of pages no longer in use.

The Second-Chance Algo
Pages are sorted in FIFO
If page fault occurs at time 20 , start from top

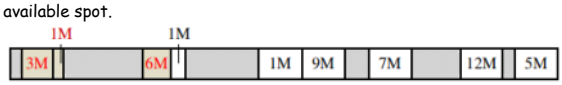
Clock Algo
Picture clock with frames at each point. Start at 1
If 1 value of use (R) is 1, then change it to 0 and move to then next. If it is 0 then replace with incoming page and set that frame value of R to 1

Swapping System

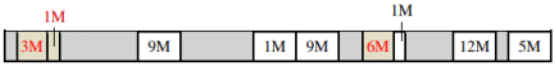


Different memory management algorithms:

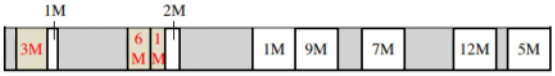
- First-fit (FF)
  In this algorithm, the different memory requests are added to the first available spot.



- Best fit(BF)
  This algorithm inserts the memory request to the spot that is most similar in size.



- Next fit (NF)
  This algorithm scans from where it last left off and searches for the next available spot.



Q2 c
A logical address consists of bits of page # and offset. A physical address consists of bits of Frame # and offset

Im a simple paging system with a say 16bit logical and physical address and page size of 1024
Eg. 0000010111011110

First separate the page # offset with the offset

So, since the page size is 1024 bytes, we need an offset field of 10 bits so 6 bits are left for the page number. Meaning a page can consist of a max 2^6 = 64 64 pages of 1K bytes each.

With 6 bit page number, we reference the page table to see which frame number is being called.

Internal fragmentation occurs when the last memory allocation unit is not full
External fragmentation occurs when space is wasted between two allocation units

Q4
i)
What are the operations you can perform on a semaphore?
Firstly, a semaphore is an integer variable used to count the number of wakeups saved for future use.

Down - if semaphore value is more than 0, decrement and continue. If value is 0, put process to sleep.

Up- increment value of semaphore.

ii)
User-level Thread: All the work of thread management is done by application and kernel is not aware of existence of threads
Advantages:
Thread switching does not require kernel mode privileges.
Scheduling can be application specific
ULTs can run on any OS

Q3 b
A virtual memory system uses page size of 4096 (4K) bytes. Virtual address consists of 32 bits.
Page size = 2^12 bit offset within a page
Virtual address size = 32 bits => range of virtual addresses 0 to 2^32 -1

i. How many entries would a page table in this system contain.
   # of entries = 2^32/2^12 = 2^20

ii. How would the memory management unit convert a virtual address into an address in real memory
   Using the page bits as a reference in a page table to find where the corresponding location in memory is. When the page is found the bits are added onto the bit offset

C
In a virtual memory management system, under what conditions are inverted page tables needed? How is such a table used?

Inverted tables are needed in cases where there are a lot of pages and a need to save on space.
Structure is called inverted because it indexes page table entries by frame number instead of virtual page number

For a physical memory size of 2^m frames the inverted page contains 2^m entries so the ith entry refers to frame i. Virtual address includes n-bit page number with n>m. Hash function maps the n-bit page number to an m-bit quantity which is used to index into the inverted page table.

Contiguous Allocation
The file is stored in consecutive blocks one after the other. On a disk with 1KB blocks, a 50 Kb file will be allocated 50 consecutive times.
Each file begins at the start of a new block so if file A takes up 3 1/2 blocks, 1/2 block is wasted.
Advantage:
Simple to implement cause keeping track of where a file's blocks is just to remember 2 numbers: disk address of the first block and number of blocks in file

Read performance is excellent cause entire file can be read from disk in a

Kernel is not aware of existence of threads

Advantages:
Thread switching does not require kernel mode privileges.
Scheduling can be application specific
ULTs can run on any OS

Disadvantages:
In a typical OS, many system calls are blocking
In a pure ULT strategy, a multithreaded app cannot take advantage of multiprocessing.

Kernel-level Thread: All the work of thread management is done by kernel.

Advantages:
Kernel can simultaneously schedule multiple threads from same process on multiple processors
If one thread in process is blocked, kernel can schedule another thread of same process
Kernel routines can be multithreaded

Disadvantages:
Transfer of control from on thread to another within same process requires mode switch to kernel

iii)
To enter critical region:
registerLock = lock
Lock = 1
While registerLock is 1:
RegisterLock = lock
Lock = 1
Return

To leave a critical region:
Lock = 0
Return

iv)
What is defect with both Petersons solution and above solution
Defect - time wasted busy waiting

Simple to implement cause keeping track of where a file's blocks is just to remember 2 numbers: disk address of the first block and number of blocks in file

Read performance is excellent cause entire file can be read from disk in a single operation.

Disadvantage:
Over time, disk becomes fragmented.
Used in CD-ROMs

Linked-List Allocation
Keeps each one as a linked list of disk blocks. First word of each block is used as a pointer to the next, rest of block is for data. Every disk block can be used in this method. No space is lost to fragmentation. Directory entry can merely store the address of first block

Disadvantage:
Random access is extremely slow
Amount of data storage in a block is no longer a power of 2 because pointer takes up a few bytes. This is less efficient cause many programs read and write in blocks whose size is a power of 2.

Linked-List Allocation Using a Table in Memory
Both disadvantages of linked-list allocation can be eliminated by taking pointer word from each disk block and putting it into a table in memory.
Using this organization the entire block is available for data. Random access is much easier. Even though chain must still be followed, it is entirely in memory so there is no need for disk references.
Disadvantages:
Doesn't scale well to large disks. Entire table must be in memory all the time. With a 1-TB disk and 1-KB block size table needs 1 billion entries. This will end up taking up 3GB of main memory all the time.

I node is used to keep track of which block belongs to which file

Race Condition: Where two or more processes are reading or writing some shared data and the final result depends on who runs precisely when

Producer consumer problem: Two processes share a common fixed size buffer. The producer adds to the buffer and the consumer takes out. The problem arises when the producer tries to add, but the buffer is already full. The solution is for the producer to go to sleep. Similarly if the consumer tries to remove but the buffer is empty, the consumer is put to sleep. This can lead to race conditions.

Assignment 3 Notes

Friday, 6 April 2018 4:12 AM

A journaling file system is a system that logs what it's going to do before it does it so in the event that the system crashes, upon reboot, it can look into the log, see what it was doing at that point in time and finish executing it.