**Urgent Pointer**

Cờ này để xác định dữ liệu là "khẩn cấp" tức là nâng độ ưu tiên của Segment. Những Segment có cờ Urgent Pointer được bật không phải chờ đợi cho đến khi các Segment trước đó được xử lý mà được gửi và xử lý ngay lập tức.

**ACKnowledgement**

Cờ ACK được sử dụng để xác nhận việc nhận thành công một hoặc nhiều gói tin. Nếu bên nhận nhận được gói tin thì sẽ gửi lại cho bên gửi cờ ACK=1 cho máy gửi để thông báo đã nhận thành công.

**PUSH**

Cờ PUSH giống như cờ Urgent Pointer, tồn tại để đảm bảo rằng các dữ liệu được ưu tiên và được xử lý tại nơi gửi hoặc nơi nhận

Được áp dụng khi bên truyền yêu cầu dữ liệu được gởi trực tiếp đến ứng dụng mà không cần thông qua vùng đệm.

Khi 1 máy muốn gửi dữ liệu của nó, nó tạm thời được cất trong bộ đệm TCP cho đến khi Segment đã đạt đến 1 kích thước nhất định sẽ gửi đến bên nhận.

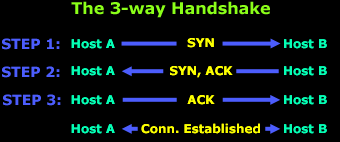
**RST (Reset)**

Cờ RST là cờ được sử dụng khi 1 Segment gửi đến mà không được dùng trong các kết nối hiện thời.

Nói cách khác, nếu bạn gửi 1 gói tin đến 1 máy chủ để thiết lập kết nối và máy chủ không có dịch vụ nào đáp ứng yêu cầu thì máy chủ sẽ gửi gói tin trả lời với cờ ACK và RST.

**(SYN) Synchronisation**

Là cờ được sử dụng thường xuyên nhất. Khi muốn thiết lập kết nối giữa 2 thiết bị với nhau thì dùng cờ SYN để yêu cầu và xác nhận kết nối. Cờ SYN được sử dụng để khởi tạo quá trình bắt tay 3 bước trong TCP.

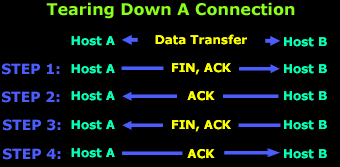


1. A gửi SYN để yêu cầu kết nối
2. B nhận SYN, B gửi SYN, ACK để xác nhận thiết lập kết nối
3. A gửi ACK xác nhận thiết lập kết nối
4. Quá trình truyền dữ liệu bắt đầu

**FIN (Finish)**

Cờ này luôn xuất hiện khi các gói dữ liệu cuối cùng được trao đổi giữa 1 kết nối. TCP hoạt hoạt động ở cả 2 hướng nên để đóng kết nối 2 bên đều gửi cờ FIN để báo rằng kết nối được đóng ở phía nó.

Thủ tục ngắt kết nối được thực hiện như sau:



1. A gửi gói FIN, ACK báo đóng kết nối, A tự đóng kết nối
2. B nhận FIN, ACK
   1. B gửi ACK báo đã nhận yêu cầu đóng kết nối
   2. Thông báo cho ứng dụng bên B ngừng gửi tin
3. B gửi FIN, ACK cho A xác nhận đã đóng kết nối bên phía mình
4. A gửi ACK báo đóng kết nối hoàn tất.

2 cờ được sử dụng trong bắt tay 3 bước và truyền dữ liệu là SYN và ACK.

**1> SYN: Các chương trình máy con (như web browser, ftp, ...) bắt đầu connection với máy chủ bằng cách gửi 1 packet với cờ "SYN" đến máy chủ:**

SYN packet này thường được gửi từ các cổng cao (1024 -65535 ) của máy con đến những cổng trong vùng thấp (1 -1023) của máy chủ. Chương trình trên máy con sẽ hỏi hệ điều hành cho 1 cổng để mở connection với máy chủ. Những cổng trong vùng này được gọi là "cổng máy con" ( client port range).

Tương tự như vậy, sẽ hỏi HĐH để nhận được quyền chờ tín hiệu trong máy chủ, vùng cổng 1-1023. Vùng cổng này được gọi là "vùng cổng dịch vụ" (service port). Ví dụ Web Server sẽ luôn chờ tín hiệu ở cổng 80 và IE sẽ connect vào cổng 80 của máy chủ.

Ngoài ra trong gói dữ liệu còn có thêm địa chỉ IP của máy con và máy chủ (cả 2)

**2. SYN/ACK: khi yêu cầu mở connection được máy chủ nhận được tại cổng đang mở, server sẽ gửi lại packet chấp nhận với 2 bit SYN và ACK:**

SYN/ACK packet được gửi ngược lại bằng cách đổi 2 IP của server và client, client IP sẽ thành IP đích và server IP sẽ thành IP bắt đầu. Tương tự như vậy, cổng cũng sẽ thay đổi, server nhận được packet ở cổng nào thì cũng sẽ dùng cổng đó để gủi lại packet vào cổng mà client đã gửi.

Server gửi lại packet này để thông báo là server đã nhận được tín hiệu và chấp nhận connection, trong trường hợp server không chấp nhận connection, thay vì SYN/ACK bits được bật, server sẽ bật bit RST/ACK (Reset Acknowledgement) và gởi ngược lại RST/ACK packet. Hoặc ICMP cổng không chấp nhận để thông báo cho client rằng yêu cầu đã bị từ chối .

Server bắt buộc phải gửi thông báo lại bởi vì TCP là chuẩn tin cậy nên nếu client không nhận được thông báo thì sẽ nghĩ rằng packet đã bị lạc và gửi lại thông báo mới.

**3> ACK Khi client nhận được SYN/ACK packet thì sẽ trả lời bằng ACK packet:**

Packet này được gởi với mục đích duy nhất báo cho máy chủ biết rằng client đã nhận được SYN/ACK packet và lúc này connection đã được thiết lập và dữ liệu sẽ bắt đầu lưu thông tự do giữa connection.

**1 Client gởi một segment đến Server với thông tin:**

SYN = 1 (Session được đồng bộ hoá)

ACK = 0 (Không có giá trị trong trường ACK, vì vậy cờ này có giá trị là 0)

Sequence Number = x, với x là một giá trị. (x là ISN của Client)

Acknowledgement Number = 0

**2 Server nhận segment này và phản hồi lại Client thông tin sau:**

SYN = 1 (Session vẫn được đồng bộ hoá)

ACK = 1 (Cờ xác nhận đã được thiết lập)

Sequence Number = y, với y là một giá trị. (y là ISN của Server)

Acknowledgement Number = x + 1 (Số Sequence Number từ **Client** được cộng thêm 1.)

**3. Client nhận segment từ Server và phản hồi đến Server các thông tin:**

SYN = 0 (Session đã được đồng bộ hoá; hơn nữa các yêu cầu không cần thiết.)

ACK = 1 (Cờ ack thiết lập response tới SYN từ segment trước)

Sequence Number = x + 1 (Đây là thứ tự trình tự tiếp theo trong chuỗi)

Acknowledgement Number = y + 1 (Số sequence number từ Server được cộng thêm 1)