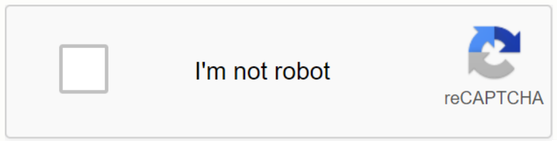

Continue

Sliding window protocol

Error detection protocol type in data link layer and transport layer for TCP This article includes a list of general references, but is not largely verified because it does not have enough corresponding online appointments. Please help improve this article by entering more accurate citations. (August 2010) In 1987, the Government of China decided to delete this template message. A sliding window protocol is a feature of package-based data transmission protocols. Sliding window protocols are used when reliable delivery is required in package order, such as in the data link layer (OSI layer 2) as well as in the Transmission Control Protocol (TCP). They are also used to improve efficiency when the channel can include high latency. Package-based systems are based on the idea of sending a batch of data, the package, along with additional data that allows the receiver to make sure it was received correctly, perhaps a checksum. When the receiver verifies the data, it sends a recognition signal, or ACK, back to the sender to indicate that it can send the next package. In a simple automatic repeat request protocol (ARQ), the sender stops after each package and waits for the receiver in ACK. This ensures that packages reach the correct order, since they can only be sent one at a time. The time it takes to receive the ACK signal can represent a significant amount of time compared to the time needed to send the package. In this case, the overall performance may be much lower than theoretically possible. To do this, sliding window protocols allow you to send a selected number of packages, the window, without having to wait for an ACK. Each package receives a sequence number and the ACK re-submits that number. The protocol tracks packages that have been acked, and when they are received, it sends more packages. This way, the window slides along the sequence of packages that make the transfer. Sliding windows are a key part of many protocols. It is a key part of the TCP protocol, which inherently allows packages to arrive out of service, and is also found in many file transfer protocols such as UUCP-g and ZMODEM as a way to improve efficiency compared to non-windowed protocols such as XMODEM. Basic concept Conceptually, each portion of the transmission (packages in most layers of data link, but bytes in TCP) is assigned a unique consecutive sequence number, and the receiver uses the numbers to place received packages in the correct order, discarding duplicate packages and identifying the missing. The problem with this is that there is no limit on the size of the sequence number that may be required. By placing limits on the number of packages that can be transmitted or received at any given time, a sliding allows you to communicate an unlimited number of packages using fixed-size sequence numbers. The term window on the transmitter side represents the logical limit of the total number of packages yet to be recognized by the receiver. The receiver reports transmitter in each recognition package of the maximum current size of the receiver buffer (window limit). The TCP header uses a 16-bit field to report the size of the receiver window to the sender. Therefore, the largest window that can be used is 216 = 64 kilobytes. In slow start mode, the transmitter begins with a low package count and increases the number of packages on each transmission after receiving receiver recognition packages. For each ack package received, the window slides through a package (logically) to transmit a new package. When the window threshold is reached, the transmitter sends a package for a received ack package. If the window limit is 10 packages, then in slow start mode the transmitter can start transmitting a package followed by two packages (before transmitting two packages, you must receive an ack package), followed by three packages and so on up to 10 packages. But after reaching 10 packages, additional transmissions are restricted to a package transmitted by an ack package received. In a simulation this appears as if the window moves by a package distance for each ack package received. On the receiving side also the window moves a package for each package received. The sliding window method ensures that traffic congestion on the network is avoided. The application layer will still be offering data for transmission to TCP without worrying about network traffic congestion problems such as sender and receiver tcp side by side to implement package buffer sliding windows. Window size can vary dynamically depending on network traffic. For the maximum possible performance, it is important that the transmitter is not forced to stop sending by the sliding windows protocol before a time of delay back and forth (RTT). The limit of the amount of data you can send before you stop waiting for a recognition must be greater than the communications link bandwidth product. Other than this, the protocol will limit the effective bandwidth of the link. Motivation In any communication protocol based on automatic repetition request for error control, the receiver must recognize the packages received. If the transmitter does not receive recognition within a reasonable time frame, re-send the data. A transmitter that does not receive an acknowledgment cannot know if the receiver actually received the package; may have been lost or damaged in transmission. If the bug detection mechanism reveals corruption, the package will be ignored by the receiver and the receiver will send a negative or duplicate recognition. The receiver can also be configured not to send any recognition at all. Similarly, the receiver is usually uncertain about whether they are receiving their accolades. Recognition may have been sent, but it was lost or corrupted in the medium of In this case, the receiver must recognize the broadcast to prevent the data from being continuously resented, but otherwise they must ignore them. to ignore them. Operation The transmitter and receiver have a current sequence number nt and nr, respectively. They also have a wt and wr window size. Window sizes may vary, but in simpler implementations they are fixed. The window size must be greater than zero for any progress to be made. As typically implemented, NT is the next package to be transmitted, i.e. the sequence number of the first package has not yet been transmitted. Similarly, nr is the first package not yet received. Both numbers are increasing monotonously over time; only ever increase. The receiver can also track the highest sequence number yet received; the variable is one more than the sequence number of the highest sequence number received. For simple receivers that only accept packages in order (wr = 1), this is the same as nr, but may be larger if wr > 1. Note the distinction: All packages below nr have been received, no packages have been received above us, and between NR and NS, some packages have been received. When the receiver receives a package, it properly updates its variables and transmits a recognition with the new nr. The transmitter tracks the highest recognition na has received. The transmitter knows that all packages up, but not including na have been received, but it is uncertain about the packages between na and ns; that is, na ≤ nr ≤ us. Sequence numbers always obey the rule that na ≤ nr ≤ us ≤ nt ≤ na + wt. That is: na ≤ nr: The highest recognition received by the transmitter cannot be higher than the highest recognized by the receiver. NR ≤: The fully received package lapse cannot be extended beyond the end of partially received packages. NS ≤: The highest package received cannot be higher than ≤ the highest package sent. Transmission operation Whenever the transmitter has data to send, it can transmit up to wt packages before the last na recognition. That is, you can transmit the package number as long as nt < na+wt. In the absence of a communication error, the transmitter will soon receive recognition for all packages it has sent, leaving it equal to nt. If this does not happen after a reasonable delay, the transmitter must relay the packages between na and nt. Techniques for defining reasonable delay can be extremely elaborate, but only affect efficiency; the basic reliability of the sliding window protocol does not depend on the details. Receiver operation Every time an x numbered package is received, the receiver checks if it falls in the reception window, nr ≤ x < nr+wr. (Simpler receivers should only track a value nr=ns.) If it falls within the window, the receiver > accepts it. previous packages have been received. [1] If x≥, the latter is updated to ns=x+1. If the package number is not inside the reception window, the receiver discards it and does not modify nr or ns. Whether the package has been accepted or not, the receiver transmits a recognition that contains the current nr. (Recognition may also include information about additional packages received between nr or ns, but this only helps efficiency.) Note that it does not make sense to have the reception window larger than the wt transmission window, because there is no need to worry about receiving a package that will never be transmitted; the useful range is 1 ≤ wr ≤ wt. The sequence number range required the main article: serial number arithmetic sequence number modulo 4, with wr=1. Initially, nt=nr=0 So far, the protocol has been described as if the sequence numbers are of unlimited size, increasing. However, instead of transmitting the full sequence number x in messages, it is possible to transmit only x mod N, for some finite N. (N is usually a power of 2.) For example, the transmitter will only receive accolades in the na-to-nt range, inclusive. Since it guarantees that nt−na ≤ wt, there are at most wt+1 possible sequence numbers that could arrive at any given time. Therefore, the transmitter can unequivocally decode the sequence number as long as N > wt. A stronger restriction is imposed by the receiver. The functioning of the protocol depends on the receiver being able to reliably distinguish new packages (which must be accepted and processed) from broadcasts of old packages (which must be discarded, and the latest recognition transmitted). This can be done taking into account the size of the transmitter window. After receiving an x numbered package, the receiver knows that x < na+wt, so that na > x−wt. In this way, x−wt numbered packages will never be relayed again. The lowest sequence number we will ever receive in the future is ns−wt The receiver also knows that the transmitter's na cannot be higher than the highest recognition ever sent, which is nr. So the highest sequence number we could see is nr+wt ≤ ns+wt. Thus, there are 2wt different sequence numbers that the receiver can receive at once. So, it might seem like we should have N ≥ 2wt. However, the actual limit is lower. The additional idea is that the receiver does not need to distinguish between sequence numbers that are too low (less than nr) or that are too high (greater than or equal to ns+wr). In any case, the receiver ignores the package except to relay a recognition. Therefore, it is only necessary that N ≥ wt+wr. Because it is common to have wr<wt (for example, see Go-Back-N below), this can allow larger wt within a N. Fixed examples The simplest sliding window: stop-and-wait Although it is commonly distinguished from the sliding window protocol, the ARQ protocol is actually the easiest possible implementation of it. The transmission window is 1 pack, and the reception window is 1 1 Therefore, N = 2 possible sequence numbers (conveniently represented by a single bit) are required. Example of ambiguity The transmitter alternately sends packages marked as strange and even. Thanks also say weird and even. Suppose the transmitter, having sent a strange package, did not expect a strange recognition, and instead immediately sent the next even package. You could then get a recognition by saying waiting for a strange package below. That would leave the transmitter in a quadary: did the receiver receive both packages, or either? Go-Back-N Go-Back-N ARQ is the sliding window protocol with wt>1, but a fixed wr=1. The receiver refuses to accept any package, but the next in sequence. If a package is lost in transit, the following packages are ignored until the missing package is relayed, a minimum loss of a round-trip time. For this reason, it is inefficient in the links that suffer the frequent loss of packages. Example of ambiguity Suppose we are using a 3-bit sequence number, such as it is typical for HDLC. This gives N=23=8. Since wr=1, we must limit wt≤7. This is because, after transmitting 7 packages, there are 8 possible results: Anywhere from 0 to 7 packages could have been successfully received. These are 8 possibilities, and the transmitter needs enough information in recognition to distinguish them all. If the transmitter sent 8 packages without waiting for recognition, it could find itself in a similar quandary to the stop-and-wait case: does recognition mean the 8 packages were successfully received, or any of them? Selective repetition The most general case of the sliding window protocol is selective repeat ARQ. This requires a much more capable receiver, who can accept packages with sequence numbers higher than the current nr and store them until the vacuum is filled. The advantage, however, is that it is not necessary to discard the correct data for a round trip time before the transmitter can be informed that a broadcast is required. Therefore, it is preferred for links with low reliability and/or a high bandwidth product. The window size should only be larger than the number of consecutive lost packages that can be tolerated. Therefore, small values are popular; wr=2 is common. Example ambiguity The extremely popular HDLC protocol uses a 3-bit sequence number, and has an optional layout for selective repetition. However, if selective repetition is to be used, the requirement must be maintained ≤ nt+nr 8; if wr is increased to 3, wt should be reduced to 6. Let's say wr =2, but an unchanged transmitter is used with wt =7, as is normally used with the GO-back-N variant of HDLC. Let's say the receiver starts with nr =ns =0. Now suppose the receiver sees the next series of packages (all modulo 8): 0 1 2 3 4 5 6 0 Because wr =2, the receiver will accept and store the final package 0 (thinking it is package 8 of the series), while requesting a broadcast package 7. However, the transmitter may not have received any recognition and relayed package 0. In the latter case, the receiver would accept the wrong package as package 8. The solution is for the transmitter to limit wt ≤6. With this restriction, the receiver knows that if all the recognitions were lost, the transmitter would have stopped after package 5. When receiving package 6, the receiver may infer that the transmitter received recognition of package 0 (na ≥1 of the transmitter), and therefore the next numbered package 0 must be package 8. Extensions There are many ways to expand the protocol: the examples above assume that packages are never rearranged in transmission; they can be lost in transit (troubleshooting makes corruption equivalent to loss), but they will never appear out of service. The protocol can be extended to support the reordering of the package, provided that the distance can be delimited; the sequence number modulus N must be extended by the maximum deordering distance. It is possible not to recognize all packages, as long as a recognition is sent eventually if there is a pause. For example, TCP normally recognizes every second package. It is common to immediately inform the transmitter if a gap is detected in the package sequence. HDLC has a special REJ package (reject) for it. The sizes of the transmission and reception window can be changed during communication, as long as their sum is kept within the N limit. In particular: It is common to reduce the size of the transmission window to slow the transmission to match the speed of the link, avoiding saturation or congestion. A common simplification of selective repetition is the so-called SREJ-REJ ARQ. This works with wr=2 packages and buffers after a vacuum, but only allows for a single lost package; while you wait for this package, wr=1 and if a second package is lost, no more packages will be lost. This gives most of the performance benefit of the complete selective repetition protocol, with easier implementation. See also Federal Standard 1037C Compound TCP Serial number arithmetic TCP Fast Open References ^ Peterson, Larry L. & Davie, Bruce S. Computer Networks: A Systems Approach, Morgan Kaufmann, 2000. ISBN 1-55860-577-0 Comer, Douglas E. Internet Work with TCP/IP, Volume 1: Principles, Protocols and Architecture, Prentice Hall, 1995. ISBN 0-13-216987-8 Peterson, Larry L. & Davie, Bruce S. Computer Networks: a Systems Approach, Morgan Kaufmann, 2000. ISBN 1558605142 External Links RFC 1323 – TCP Extensions for High Performance TCP Window Scaling and Broken Routers, Sliding Window Demo 2004 (Flash Required) Recovered From