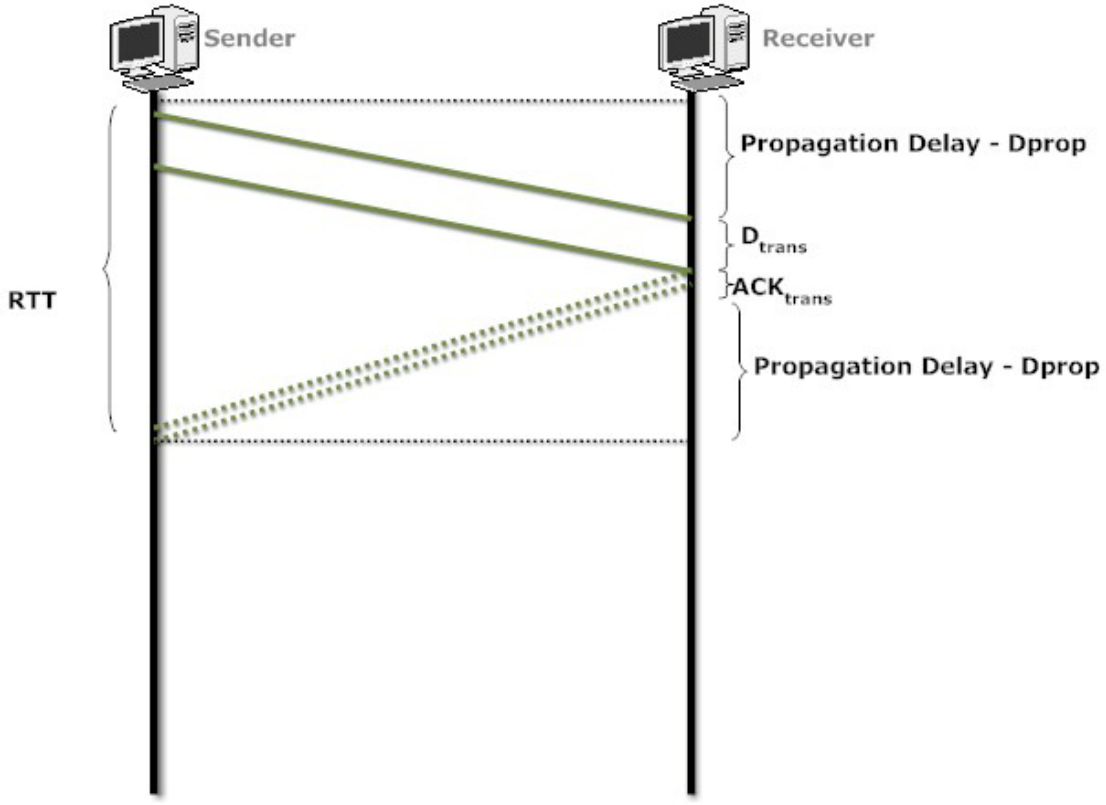
I. Παράδειγμα 1: Απόδοση TCP με παράθυρο αποστολέα = 1

a. Ο μηχανισμός όπως έχει περιγραφεί ως τώρα στέλνει μόνο ένα πακέτο και σταματάει να μεταδίδει έως ότου πάρει το ack του πακέτου αυτού (λειτουργία stop-and-wait). Αυτός ο τρόπος όμως δεν είναι αποδοτικός.

b. Προτού παρουσιαστεί παράδειγμα, γίνεται η παρουσίαση κλασσικού διαγράμματος για TCP και επεξήγηση propagation delay και transmission delay.



Σχήμα 1: TCP αποστολέας με παράθυρο αποστολέα = 1

Propagation Delay – Καθυστέρηση διάδοσης σήματος στο φυσικό μέσο – χρόνος που απαιτείται για να φθάσει στον παραλήπτη το πρώτο bit της μετάδοσης.

Transmission Delay (D_{trans}) – Χρόνος που χρειάζεται για να «γραφτεί» όλο το πακέτο στο κανάλι – να εισαχθεί και το τελευταίο bit για μετάδοση. Εξαρτάται από το πόσο γρήγορα μπορούν να γράφονται bits στο μέσο. Αλλιώς αναφερόμαστε σε αυτό το χρόνο και σαν χρόνο που χρειάζεται για να μεταδοθεί το πακέτο bit προς bit στο μέσο (καλώδιο ή ασύρματο κανάλι).

Ack_{trans} – Χρόνος που χρειάζεται για να μεταδοθεί το ACK στο μέσο.

Round Trip Time (RTT) – Ο χρόνος που απαιτείται για να σταλεί το πακέτο συν το χρόνο που χρειάζεται για να ληφθεί ACK του πακέτου.

Throughput (ρυθμαπόδοση) – Ρυθμός με τον οποίο μπορούμε να στέλνουμε δεδομένα στο μέσο. Δεδομένων όλων των κλασσικών καθυστερήσεων (διάδοσης, μετάδοσης, επεξεργασίας – processing delay, ενταμίευσης σε ουρά – queuing delay), πόσο γρήγορα μπορούμε να μεταδίδουμε πληροφορία στο μέσο. Πόσα bits ανά μονάδα χρόνου.

Utilization (Χρησιμοποίηση του μέσου μετάδοσης): Κλάσμα του χρόνου που ο αποστολέας μετέδιδε δεδομένα στο κανάλι σε σχέση με το συνολικό χρόνο που χρειάζεται για να ολοκληρωθεί η μετάδοση.

c. Σενάριο: δύο τερματικά κατά μήκος ενός μονοπατιού στο internet. Υπόθεση για μηδενικό queuing και processing delay καθώς και για $Ack_{trans} = 0$. Έστω επίσης μέγεθος πακέτου L= 1000bytes, χωρητικότητα μονοπατιού R= 1Gps και καθυστέρηση διάδοσης Dprop = 30msec.

Ο χρόνος για να μεταδοθεί το πακέτο στο μέσο είναι: $D_{trans} = \frac{L}{R} = \frac{1000*8}{10^9} = 0{,}008 msec$.

Ο συνολικός χρόνος που χρειάζεται για να ολοκληρωθεί η μετάδοση του πακέτου είναι:

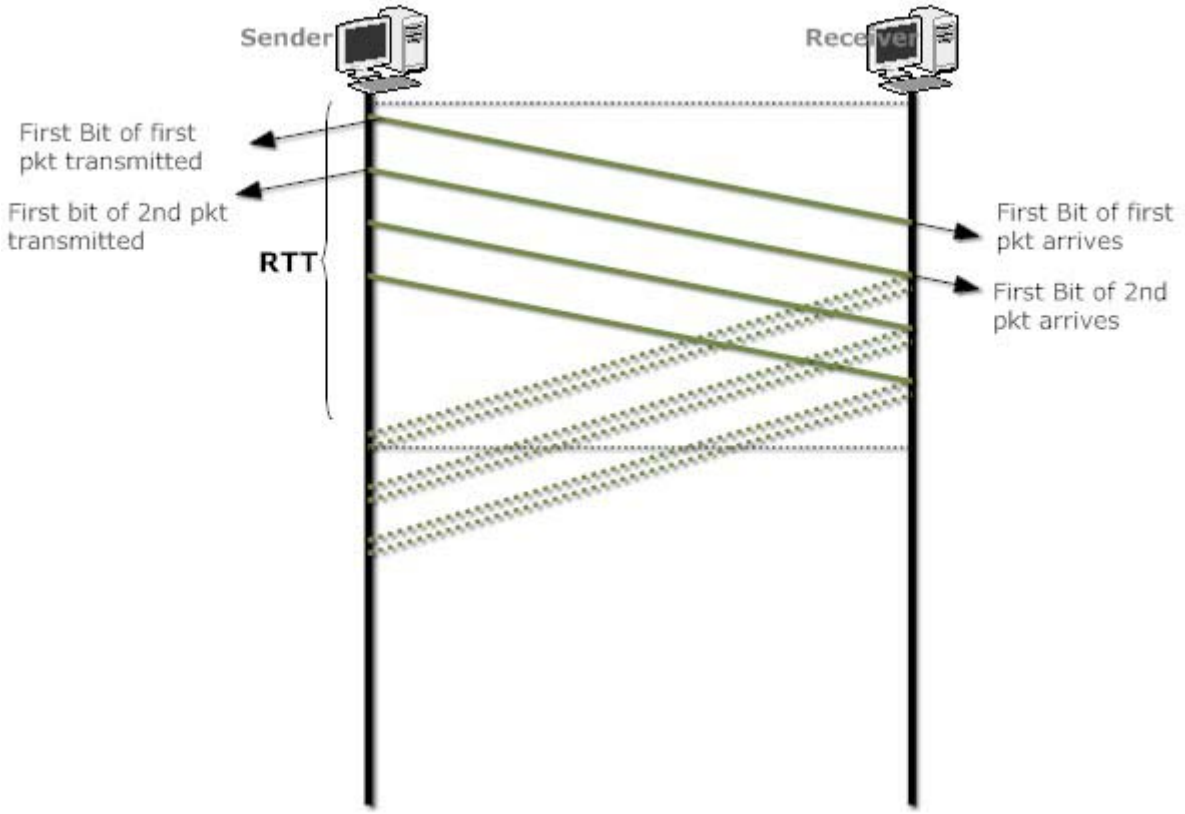
$D_{trans} + ACK_{trans}^{ACK_{trans}=0} + 2 * Dprop = 30{,}008$

Συνεπώς, το utilization του καναλιού είναι U = $\frac{0{,}008}{30{,}008} = 0{,}00027$.

Συνολικά το throughput που επιτυγχάνεται περνώντας από ένα μονοπάτι με χωρητικότητα 1Gbps είναι:
Throughput = Capacity*Utilization = 1Gbps*U=267Kbps.

II. Παράδειγμα 2: Κίνητρο για σωλήνωση μεταδόσεων – παράδειγμα με περισσότερες μεταδόσεις χωρίς αναμονή για επιβεβαίωση

Έστω ότι πραγματοποιούνται 3 μεταδόσεις πακέτων ταυτόχρονα προτού ληφθεί το πρώτο ACK.



Σχήμα 2: Αποστολέας TCP με παράθυρο μετάδοσης = 3

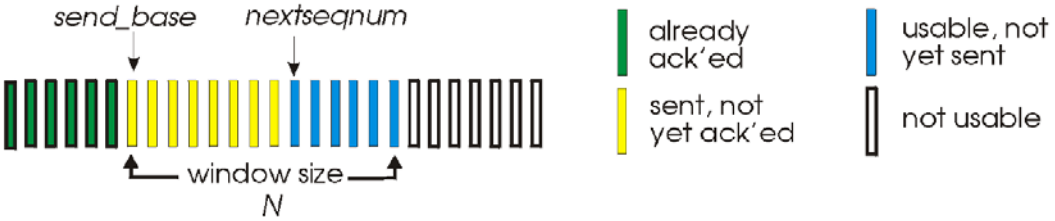
Τώρα σε χρόνο 1 RTT = $D_{trans} + ACK_{trans}^{ACK_{trans}=0} + 2 * Dprop = 30{,}008$ γίνεται η μετάδοση 3 πακέτων αντί για 1 άρα το utilization κατά παρόμοιο τρόπο θα είναι:

U = $\frac{3*0{,}008}{30{,}008} = 3*0{,}00027$ άρα έχει τριπλασιαστεί.

Τα τρία πακέτα τα οποία μεταδίδονται χωρίς αναμονή, έως ότου ληφθεί το πρώτο ACK, λέμε ότι μεταδίδονται με τη χρήση σωλήνωσης ή αλλιώς ότι έχουμε pipelined transmissions.

III. Επέκταση αξιοπιστίας για TCP με παράθυρο αποστολέα > 1

- Με την προηγούμενη τεχνική είναι σαν να έχουμε αυξήσει το πλάτος της σωλήνωσης μέσα από την οποία διέρχονται πακέτα γι' αυτό και αυτή η τεχνική ονομάζεται και σωλήνωση μεταδόσεων.
- Οι pipelined μεταδόσεις περιπλέκουν το χειρισμό των απωλειών πακέτων καθώς τώρα μπορεί να έχει χαθεί οποιοδήποτε και οσαδήποτε από όσα στέλνονται ταυτόχρονα.
- Βασικές δομικές μονάδες
  - Ας υποτεθεί ότι ο αποστολέας χρησιμοποιεί ένα παράθυρο μεγέθους N. Το N με άλλα λόγια παριστάνει τον αριθμό των πακέτων τα οποία έχουν μεταδοθεί και δεν έχουν γίνει ACK.
  - Αύξηση του εύρους των αριθμών ακολουθίας
  - Χρήση ενταμιευτών στον αποστολέα και στον παραλήπτη. Γιατί? Τι μέγεθος έχουν? Εφόσον ο αποστολέας μπορεί να στείλει π.χ. το πολύ N πακέτα ταυτόχρονα χωρίς να πάρει επιβεβαίωση, θα πρέπει να κρατάει αυτά τα N πακέτα κάπου αποθηκευμένα έως ότου έρθει επιβεβαίωση γιατί μπορεί να χρειαστεί να τα ξαναστείλει (sender buffer). Αυτά τα N πακέτα στον παραλήπτη μπορεί να μη φτάσουν στη σειρά. Μπορεί πχ να έρθει το $4^{ο}$ πακέτο, μετά το $7^{ο}$ το $1^{ο}$ κ.ο.κ. Το πρωτόκολλο TCP υλοποιεί υπηρεσία in-order-delivery στο υψηλότερο επίπεδο του δικτύου. Αυτό σημαίνει ότι στο επίπεδο που είναι αμέσως μετά από αυτό θα παραδίδει τα πακέτα με τη σωστή σειρά, από το 1 έως το N. Όσο λοιπόν παραλαμβάνει πακέτα χρειάζεται να τα αποθηκεύει κάπου έως ότου έρθουν πακέτα που καλύπτουν πιθανά κενά από αριθμούς ακολουθίας (receiver buffer).
  - Με βάση την αρχική υπόθεση ότι ο αποστολέας χρηστιμοποιεί ένα παράθυρο μεγέθους N, σε κάθε μετάδοση μπορούν να σταλούν κάποια ακόμα πακέτα χωρίς να περιμένει επιβεβαίωση. Αυτός ο έλεγχος υλοποιείται με τη χρήση ενός κυλιόμενου παραθύρου πάνω από τους διαθέσιμους αριθμούς ακολουθίας.



Σχήμα 3: Διαχείριση παραθύρου αποστολέας

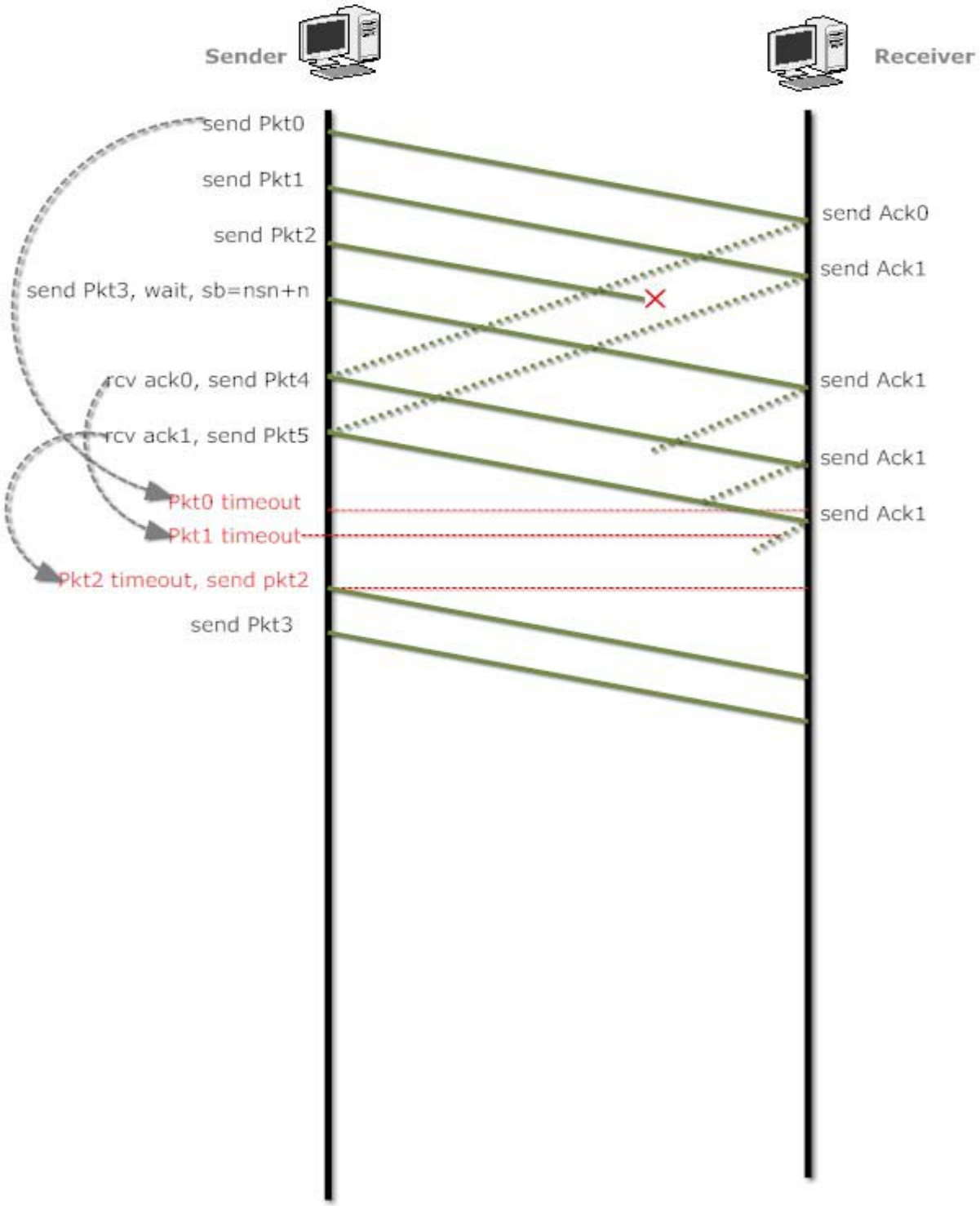
Όπως φαίνεται και στο σχήμα 3, τα σημειωμένο με πράσινο πακέτα έχουν σταλεί και έχει ληφθεί το ACK τους. Το ενεργό παράθυρο περιλαμβάνει τα κίτρινα και γαλάζια. Τα κίτρινα έχουν σταλεί αλλά δεν έχει πάρει ακόμα επιβεβαίωση ο αποστολέας. Οι παράμετροι send_base (sb) και nextseqnum (nsn) είναι μεταβαλλόμενες (αυξάνονται) δεν είναι σταθερές. Κάθε φορά ανάλογα με το γεγονός η τιμή τους αυξάνεται κατά 1.

- Διαχείριση παραθύρου στον αποστολέα: Περιληπτικά για τον sender
  - Αποστολή πακέτου: Αν sb=nsn είναι το πρώτο πακέτο του παραθύρου και αρχίζει ο timer. Όποτε στέλνεται ένα πακέτο, καταναλώνεται και ένας διαθέσιμος αριθμός από το παράθυρο άρα μετακινείται το δεξί όριο, nsn++. Όταν γίνει nsn=base+n, έχει εξαντληθεί το παράθυρο.
  - Αν συμβεί timeout για τον τρέχων counter, ξαναστέλνονται όσα πακέτα δεν έχουν γίνει ACK. Πακέτα με αριθμούς ακολουθίας από sb έως sb+nsn-1.
  - Αν παραλάβει ACK το αριστερό τμήμα του παραθύρου προχωράει κατά ένα, sb=sb+1. Όταν γίνει sb=nsn, τότε όλο το παράθυρο έχει γίνει ACK και επομένως

σταματάει να μετράει τον retransmission counter. Αν ο sb δεν έχει προχωρήσει αρκετά έτσι ώστε να γίνει ίσος με το nsn, τότε ο retransmission counter ξαναρχίζει για το τρέχων πακέτο.

- Ο παραλήπτης δουλεύει πιο απλά. Αν το πακέτο είναι αυτό που περιμένει στέλνει ACK το οποίο περιλαμβάνει τον επόμενο αριθμό ακολουθίας που περιμένει. Αν όχι στέλνει τον επόμενο αριθμό ακολουθίας από το τελευταίο πακέτο που έχει πάρει in-order.

IV. Παράδειγμα 3: Απώλεια πακέτου από ένα παράθυρο με μέγεθος 4



Σχήμα 4: Απώλεια πακέτου από ένα παράθυρο με μέγεθος 4

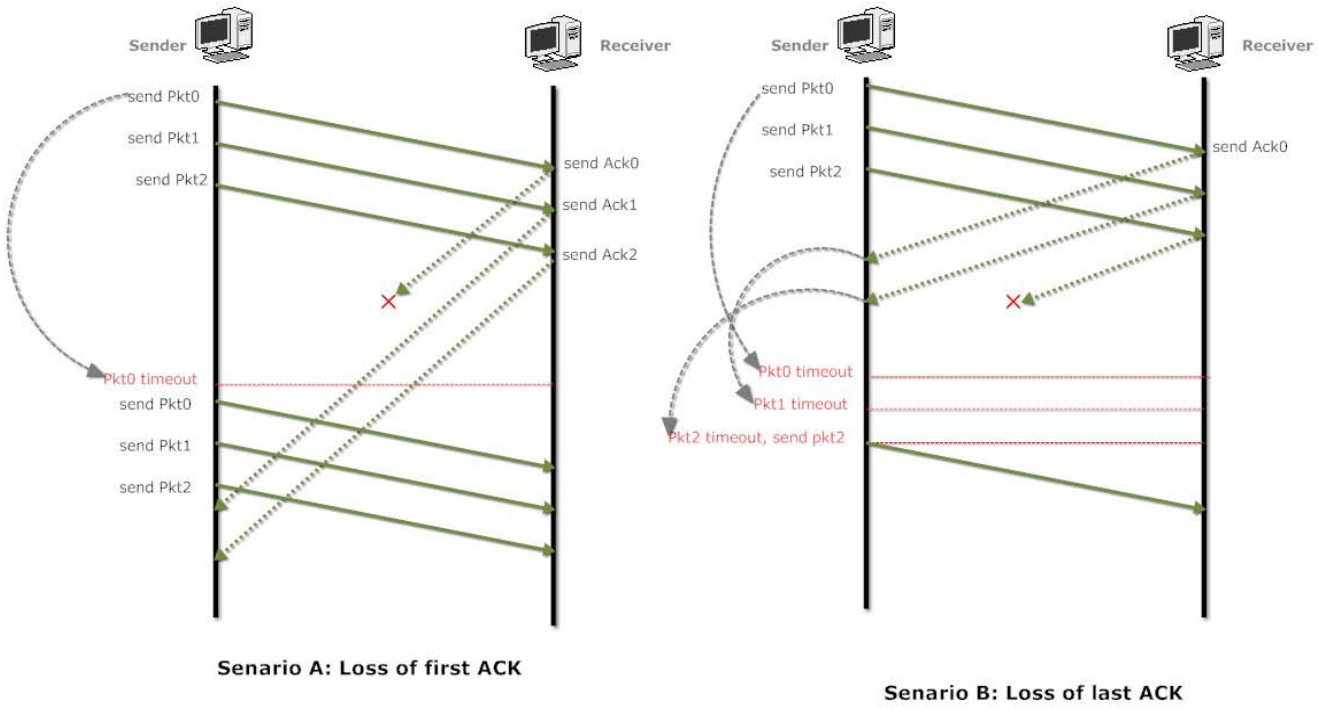
Στο παράδειγμα του σχήματος 4 παρουσιάζεται μια κλασσική περίπτωση απώλειας ενός τυχαίου πακέτου από το τρέχον παράθυρο. Εδώ ο αποστολέας στέλνει 4 πακέτα ταυτόχρονα, pkt0 έως pkt3. Ο retransmission counter γίνεται set όταν στέλνεται το πρώτο πακέτο κάθε παραθύρου και όποτε λαμβάνεται μία επιβεβαίωση. Αρχικά, μόλις στείλει το pkt0 θέτει και το pkt0 timeout (πρώτη κόκκινη γραμμή στο σχήμα 4). Το πακέτο pkt0 φτάνει χωρίς λάθη στον παραλήπτη, ο οποίος το επιβεβαιώνει στέλνοντας ack0.
Ο sender μόλις πάρει αυτό το ack προχωράει το παράθυρο κατά ένα (send base = send base + 1 = 0+1 =1) και θέτει τον retransmission counter για το επόμενο πακέτο pkt1. Επιβεβαιώνεται και αυτό και συνεπώς γίνονται οι ίδιες ενέργειες: send base = send base +1 = 1 + 1 = 2 και set(retransmission counter) για pkt2.

Το πακέτο pkt2 όμως όπως φαίνεται στο σχήμα 4 χάνεται άρα μόλις λήξει ο retransmission counter που είχε θέσει ο αποστολέας (pkt2 timeout) ο αποστολέας ξαναστέλνει όλα τα πακέτα του τρέχοντος παραθύρου από send base έως next sequence number άρα ξαναστέλνει τα πακέτα 2 και 3.

V. Παράδειγμα 4: Απώλεια τυχαίου πακέτου εν μέσω παραθύρου και υπολογισμός ελάχιστων και μέγιστων αναγκαίων επαναμεταδόσεων

Η συσκευή Α στέλνει ένα αρχείο μεγέθους Μ bits σε μία συσκευή Β χρησιμοποιώντας TCP. Το κάθε πακέτο έχει μέγεθος p. Όλα τα πακέτα λαμβάνονται στη σωστή σειρά χωρίς κανένα λάθος από τον Β. Ας υποθέσουμε όμως ότι ακριβώς ένα ACK για ένα από τα παραπάνω data packets χάνεται, αλλά όλα τα υπόλοιπα φτάνουν σωστά στον Α πριν το δικό τους timeout. Ποιός είναι ο μέγιστος και ποιός ο ελάχιστος αριθμός data packets που θα ξαναστείλει ο Α και γιατί?

Έστω ότι το παράθυρο που χρησιμοποιεί η συσκευή Α είναι 3 πακέτα. Αυτό σημαίνει ότι μπορεί να έχει ανά πάσα στιγμή το πολύ 3 πακέτα που έχουν ήδη μεταδοθεί χωρίς να έχει λάβει επιβεβαίωση για κανένα από αυτά. Ο μέγιστος και ο ελάχιστος αριθμός πακέτων που θα επαναμεταδοθούν εξαρτάται από το ποια επιβεβαίωση θα χαθεί. Προφανώς όσο πιο πρώιμη επιβεβαίωση χαθεί τόσο πιο πολλά πακέτα θα χρειαστεί να επαναμεταδοθούν. Όσο πιο πρόσφατη επιβεβαίωση χαθεί τόσο λιγότερες μεταδόσεις θα επηρρεαστούν. Ας θεωρήσουμε τα ακραία σενάρια όπου χάνεται το πρώτο ACK και το τελευταίο ACK όπως φαίνονται στο παρακάτω σχήμα, σε μία σύνδεση TCP που χρησιμοποιεί παράθυρο στο GBN μεγέθους 3.

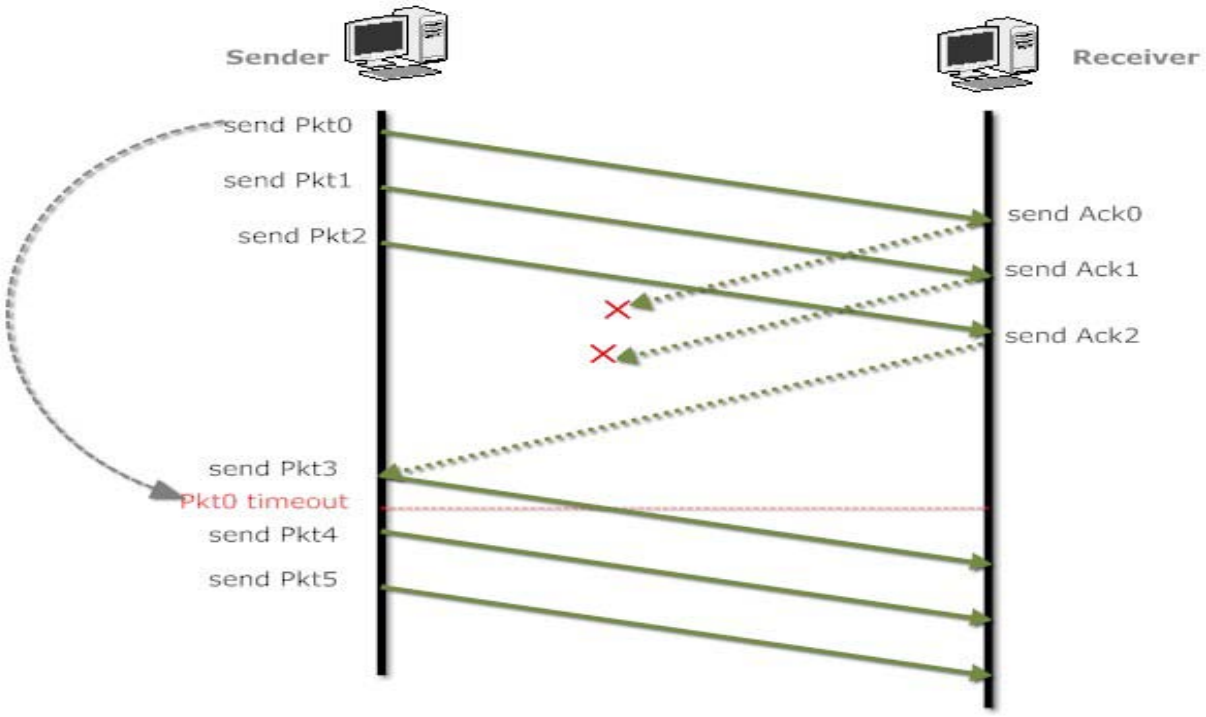


Σχήμα 5: Απώλεια τυχαίου πακέτου εν μέσω παραθύρου μετάδοσης

Στο πρώτο σενάριο που χάνεται το πρώτο ACK θα επαναμεταδοθεί όλο το παράθυρο. Στη δεύτερη περίπτωση θα επαναμεταδοθεί μόνο το τελευταίο πακέτο. Άρα ο μεγαλύτερος αριθμός πακέτων που μπορεί να χρειαστεί να επαναμεταδοθούν είναι όσα και το παράθυρο του GBN, και τα λιγότερα είναι ένα.

VI. Παράδειγμα 5: Απώλεια επιβεβαίωσης για τυχαίο πακέτο

Οι επιβεβαιώσεις που υποτίθεται ότι χρησιμοποιούμε ως τώρα είναι αθροιστικές. Εάν θεωρηθεί ότι το πεδίο της επιβεβαίωσης φέρει αριθμό πακέτου και όχι αριθμό από bytes τότε ένα ACK(3) σημαίνει ότι έχουν ληφθεί σωστά όλα τα πακέτα ως και το 3. Ας υποθέσουμε επίσης ότι το παράθυρο του αποστλέα έχει μέγεθος τρία. Θεωρείστε το παρακάτω παράδειγμα:



Σχήμα 6: Απώλεια τυχαίας επιβεβαίωσης

Σε αυτό το σενάριο οι επιβεβαιώσεις για τα πακέτα 0 και 1 χάνονται όμως πρωτού λήξει ο timeout counter για το πακέτο 0 (pkt0 timeout) ο αποστολέας δέχεται ένα ack(2). Αυτό σημαίνει πως ο παραλήπτης επιβεβαιώνει ότι έχει παραλάβει όλα τα πακέτα σωστά έως και το 2 οπότε ο αποστολέας μπορεί να προχωρήσει στην αποστολή του επόμενου παραθύρου πακέτων, 3,4,5.

VII. Παράδειγμα 6: Υπολογισμός μεγέθος παραθύρου

Έστω το σενάριο που μελετήθηκε στην Παράδειγμα 1 αυτής της αναφοράς (σχήμα 1). Πόσο θα πρέπει να είναι το μέγεθος του παραθύρου έτσι ώστε να χρησιμοποιείται το 90% της χωρητικότητας του μονοπατιού απόν τον source στον destination?

Για να χρησιμοποιείται το 90% της χωρητικότητας, τότε θα πρέπει ο αποστολέας το 90% του χρόνου που χρειάζεται για να σταλούν n πακέτα να μεταδίδει κίνηση στο δίκτυο.
Γνωρίζουμε από το προηγούμενο παράδειγμα ότι:

1 RTT = $D_{trans} + ACK_{trans} + 2 * Dprop \overset{ACK_{trans}=0}{=} 30{,}008$ και ότι L/R = 0,008.

Έτσι θέλουμε: $\frac{n * 0{,}008}{30{,}008} = 0{,}9$ όπου n το μέγεθος του παραθύρου. Συνεπώς, ο αριθμός των πακέτων που πρέπει να σταλούν ταυτόχρονα είναι περίπου 3376.