RFC 793
Aktarım Denetim Protokolü (TCP)

Düzenleyen:
Jon Postel

Çeviren:
Behice Balkaya
<comp.behicebalkaya (at) gmail.com>

Mayıs 2006

Özet

Aktarım Denetim Protokolü (TCP) paket anahtarlamalı bilgisayar ağlarındaki ve bu tür ağların birbirine bağlı sistemlerindeki konaklar arasında çok güvenilir bir konaktan konağa protokol olarak kullanılmak üzere tasarlanmıştır. Bu belge Aktarım Denetim Protokolü ve protokolün kullanıcılara veya uygulamalara arayüzünü ve kendini gerçekleyen programlar tarafından yerine getirilen işlevleri açıklar.

Konu Başlıkları

Geçmiş

1.0 İlk çeviri	Mayıs 2006	BB
Standart Özgün sürüm	Eylül 1981	JP

Sürüm Bilgileri

Information Sciences Institute
University of Southern California
4676 Admiralty Way
Marina del Rey, California 90291
tarafından
Defense Advanced Research Projects Agency
Information Processing Techniques Office
1400 Wilson Boulevard
Arlington, Virginia 22209
için hazırlanmıştır.

Açıklama İsteği: 793
Geçersiz Kıldığı RFC'ler: 761
Geçersiz Kıldığı IEN'ler: 129, 124, 112, 81, 55, 44, 40, 27, 21, 5
Güncelleyen RFC: 3168
Durumu: Standart (STD0007)

Yasal Uyarı

Önsöz

Bu belge ABD Savunma Bakanlığı (DoD) Standart Aktarım Denetim Protokolünü (Transmission Control Protocol – TCP) açıklar. Bu standardın üzerine inşa edildiği daha önceki dokuz ARPA TCP belirtimi vardır ve belirtimin metni ağırlıkla bunların metinlerinden oluşmuştur. Gerek metin gerekse kavramsal olarak bu çalışmaya pek çok kişi destek oldu. Bu sürümde çeşitli ayrıntılar arıtıldı ve mektup–sonu tampon–boyutu ayarlamaları silindi ve mektup mekanizması bir bası işlevi olarak yeniden açıklandı.

– – Jon Postel
Editör

1. Giriş

Aktarım Denetim Protokolü (TCP) paket anahtarlamalı bilgisayar ağlarındaki ve bu tür ağların birbirine bağlı sistemlerindeki konaklar arasında çok güvenilir bir konaktan konağa protokol olarak kullanılmak üzere tasarlanmıştır.

Bu belge Aktarım Denetim Protokolü ve protokolün kullanıcılara veya uygulamalara arayüzünü ve kendisini gerçekleyen programlar tarafından yerine getirilen işlevleri açıklar.

1.1. Güdülenme

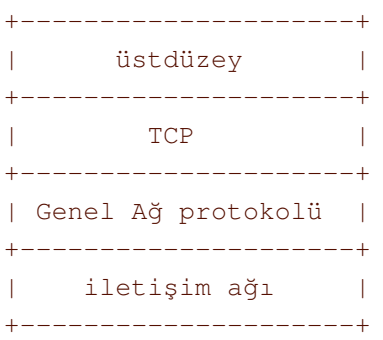
Bilgisayar iletişim sistemleri gerek orduda gerek idari teşkilatta ve gerekse sivil çevrelerde gittikçe önemi artan bir konudur. Bu belgede öncelikle askeri bilgisayarların iletişim gereksinimleri, özelikle bir tıkanıklıkta ve güvenilir olmayan bir bağlantı durumunda güvenilirliği sağlama konularına dikkat edilecektir. Ama bu sorunların çoğu sivil ve devlet sektöründe de bulunmaktadır.

Stratejik ve taktiksel olarak bilgisayar iletişim ağları esasen bilgisayarları birbirine bağlamak ve çok sayıda uygulamayı destekleyen standart süreçler arası iletişim protokollerinin tüm gereksinimlerini sağlamak için geliştirilir ve konuşlandırılır. Bu tür standartlarla ilgili gereksinimi karşılamak için Savunma Bakanlığının Araştırma ve Mühendislik Dairesi[1] Savunma Bakanlığı bünyesinde süreçler arası iletişim protokolünün standartlaştırılması temelinde burada açıklanan Aktarım Denetim Protokolünü (TCP) bildirmiştir.

TCP, çok ağlı çalışmayı destekleyen uygulamaları destekleyen protokollerin katmanlı hiyerarşisine uyum sağlamak için tasarlanmış bağlantı yönelimli, uçtan uca güvenilir iletişim sağlayan bir protokoldür. Birbirine bağlı bilgisayar iletişim ağlarına konak olarak bağlı bilgisayar çiftlerindeki süreçler arasında süreçlerarası güvenilir iletişim için gerekenleri sağlar. TCP katmanı altındaki iletişim protokollerinin güvenilirliği ile ilgili pek az önkabul yapılmıştır. TCP'nin daha düşük seviyeli protokollerden basit ve potansiyel olarak güvensiz bir verikataı hizmeti sağlayabileceği varsayılır. TCP'nin doğrudan bağlantılı ağlardan paket anahtarlamalı veya devre anahtarlamalı ağlara kadar geniş bir tayftaki iletişim sistemleri üzerinde çalışabilmesi prensip olarak mümkün olmalıdır.

TCP temel olarak ilk defa Cerf ve Kahn tarafından [1] içinde açıklanan kavramlara oturtulmuştur. TCP katmanlı protokol mimarisinde, Genel Ağ verikataı "zarfları" halinde zarflanmış değişik uzunluktaki veribölütlerini almak ve göndermek için TCP'ye bir yöntem sağlayan temel Genel Ağ Protokolünün hemen üstüne oturur. Genel Ağ verikataı farklı ağlardaki kaynak ve hedef TCP'lerin adreslenmesini sağlar. Genel Ağ protokolü ayrıca çoklu ağlar ve bağlantılı ağ geçitlerinden taşınmasını ve teslimatını sağlamak için TCP veribölütlerinin parçalanması ve yeniden birleştirilmesi ile de ilgilenir. Genel Ağ protokolü ayrıca öncelik, güvenlik sınıflandırması ve TCP veribölütlerinin bölümlere ayrılması ile ilgili bilgileri de taşır, böylece bu bilgi çok sayıda ağ arasında bir uçtan bir uca iletilebilir.

Şekil 1. Protokol Katmanları



Bu belge çoğunlukla, konak bilgisayarlarda daha yüksek seviyeli protokollerle birlikte yerleşik olan TCP gerçeklenimleri bağlamında kaleme alınmıştır. Bazı bilgisayar sistemleri ağlara bağlanmak için üzerlerinde ağa özel yazılımlara ek olarak TCP ve IP katmanlarını barındıran bilgisayarları kullanırlar. TCP belirtimi, bağlanma noktasında bile gerçeklenebilir görünen daha yüksek seviyeli protokollere bir arayüzden başka bağlanma noktası protokolü olarak da gerçeklenmeye uygun bir arayüz tanımlar (Burada bağlantı noktasından kasıt, sistemin ağa bağlantısını sağlayan bilgisayardır).

1.2. Etki Alanı

TCP, çok ağlı bir ortamda iki süreç arasında güvenilir bir iletişim sağlamak için düşünülmüştür. TCP, çoklu ağlarda ortak kullanım için iki konak arası protokol olarak tasarlanmıştır.

1.3. Belge Hakkında

Bu belgede bir TCP gerçekleniminin, gerek daha yüksek seviyeli protokollerle etkileşim gerekse diğer TCP gerçeklenimleriyle etkileşimin gerektirdiği davranışın bir belirtimi sunulmaktadır. Bu bölümün kalanında protokol arayüzlerine ve işlemlerine kısaca bir gözatılmıştır. *FELSEFE* (sayfa: 8) bölümünde TCP tasarımının felsefi temeli özetlenmiştir. *İşlevsel Belirtim* (sayfa: 12) bölümünde ise TCP veribölütlerinin biçimlerinin ayrıntıları ile oluşan çeşitli olaylarda (yeni veribölütlerin gelişi, kullanıcı çağrıları, hatalar, vs.) TCP'nin verdiği tepkilerin ayrıntıları açıklanmıştır.

1.4. Arayüzler

TCP arayüzlerinin bir tarafı Genel Ağ Protokolü gibi daha düşük seviyede bir protokole bakarken, diğer tarafı kullanıcıya veya uygulama süreçlerine bakar.

Uygulama süreçleri ve TCP arasındaki arayüz makul derecede bir ayrıntıyla ele alınmıştır. Bu arayüz bir işletim sisteminin dosyaları işlemesi için bir uygulama sürecine sağladığı çağrılara çok benzeyen bir çağrı kümesinden oluşur. Örneğin, bağlantıları kurmak ve kapamak, kurulu bağlantılar üzerinden veri göndermek ve almak için çağrılar vardır. Ayrıca TCP'nin uygulama programları ile eşzamansız haberleşmesi beklenir. TCP gerçeklenimcilerine belli bir işletim sistemine uygun arayüzlerin tasarlanması için epeyce özgürlük tanınmış olsa da, TCP/kullanıcı arayüzünde geçerli bir gerçeklenim için asgari bir işlevsellik gereklidir.

TCP ile daha düşük seviyeli protokol arasındaki arayüz aslında belirlenmemiştir, ancak iki seviyenin birbirlerine eşzamansız olarak bilgi aktarabilecekleri bir mekanizmanın olduğu varsayılır. Genel olarak, bu arayüzü daha düşük seviyedeki protokolün belirleyeceği umulur. TCP birbirine bağlı ağlardan oluşan çok genel bir ortamda çalışmak üzere tasarlanmıştır. Bu belge boyunca, daha düşük seviyeli protokolün Genel Ağ Protokolü [*2*] olduğu varsayılmıştır.

1.5. İşlemler

Evvelce belirtildiği üzere, TCP'nin birincil amacı süreç çiftleri arasında güvenilir ve güvenli olabilen mantıksal bir devre veya bağlantı hizmeti sağlamaktır. Bir az güvenilir ağ iletişim sisteminin tepesinde bu hizmeti sağlamak için aşağıdaki alanlardaki oluşumlara ihtiyaç vardır:

- Temel Veri Aktarımı
- Güvenilirlik
- Akış Denetimi
- Çoğullama
- Bağlantılar
- Öncelik ve Güvenlik

TCP'nin bu alanların her birindeki temel işlemleri aşağıdaki paragraflarda açıklanmıştır.

Temel Veri Aktarımı:

TCP, Genel Ağ sistemi üzerinden aktarım için bazı sayıları veribölütleri içinde sekizliler şeklinde paketleyen kullanıcılar arasında iki yönlü olarak sürekli bir sekizliler akımı olarak aktarabilir. Genelde, TCP verinin ne zaman durdurulacağına ve ne zaman bırakılacağına kendi kuralları dahilinde karar verir.

Bazen kullanıcılar gönderilmesi için TCP'ye teslim ettikleri verilerin tamamının iletildiğinden emin olmak isterler. İşte bu amaçla bir gitsin işlevi tanımlanmıştır. TCP'ye teslim edilmiş verinin gerçekten iletildiğinden emin olmak için gönderici tarafın bu verinin alıcı tarafa doğru itilmesi gerektiğini belirtmesi gerekir. Bir GİTsin, TCP'nin hemen bu noktada alıcıya verileri göndermesine ve teslim etmesine sebep olur. gitme noktası alıcı tarafa görünür olmayabileceği gibi gitsin işlevi bir kayıt sınırlama işaretçisi de sağlamaz.

Güvenilirlik:

TCP'nin Genel Ağ iletişim sistemi tarafından hasar verilmiş, kaybedilmiş, yinelenmiş veya sırası değiştirilerek teslim edilmiş verileri kurtarması gerekir. Bunu sağlamak için aktarılacak her sekizliye alıcı TCP tarafından olumlu bir alındı (ALN) gönderilmesini gerektiren bir sıra numarası atanır. Eğer bir zamanaşımı süresi içinde ALN alınmazsa veri yeniden aktarılır. Alıcı tarafta bu sıra numarası gönderilen veribölütlerin doğru sıraya konması (veribölütleri karışık bir sırada gelmiş olabilir) ve gelen veribölütlerinin içinden yinelenmiş olanların ayıklanması için kulanılır. Aktarılan veribölütlerinde meydana gelen hasarı saptamak için her veribölütüne, alıcı tarafta sınanması ve bozuksa veribölütünün iptal edilebilmesi için bir sağlama özeti eklenir.

TCP'ler düzgün olarak işlediği sürece ve Genel Ağ sistemi tamamen parçalanmadıkça verinin düzgün teslimatını etkileyecek hiçbir aktarım hatası olmaz.

Akış Denetimi:

TCP alıcıya gönderici tarafından gönderilmiş verinin miktarını yönetme imkanı da sağlar. Bunun için her ALN ile bir "pencere" döndürülür ve bu pencere son veribölütün başarıyla alındığını belirtmekten başka alıcıdan bir izin alınmasını gerektirmeden gönderilebilecek sekizlilerin sayısını belirten bir sıra numarası aralığı içerir.

Çoğullama:

Tek bir konak içinde birden fazla sürecin TCP iletişim oluşumlarını aynı anda kullanabilmelerini mümkün kılmak için TCP her konağa bir takım adresler ve portlar sağlar. Genel Ağ iletişim katmanındaki ağ ve konak adreslerine kademeli eklenerek bu bir soketi şekillendirir. Bir bağlantı bir soket çifti gerektirdiğinden bir soket aynı anda çok sayıda bağlantıda kullanılabilir.

Süreçlerin portlara bağlanması her konak tarafından bağımsız olarak yürütülür. Yine de, sıklıkla kullanılan süreçleri (örn, bir günlük kayıtçısı veya zaman paylaşım hizmeti) herkesin bildiği hep aynı soketlere bağlamak kullanışlılık sağlar. Bu hizmetlere böylece bilinen adreslerden erişilebilir. Diğer süreçlerin port adreslerinin kurulması ve öğrenilmesi daha anlık mekanizmaların işe karışmasını gerektirebilir.

Bağlantılar:

Yukarıda açıklanan güvenilirlik ve akış denetim mekanizmaları TCP'lerin her veri akımı için ilklendirilmesini ve belli bir durum bilgisinin sürekli sağlanmasını gerektirir. Soketleri, sıra numaralarını ve pencere boyutlarını da içererek bu bilgi bütününe bağlantı adı verilir. Her bağlantı her biri bir ucu belirten bir çift soket tarafından eşsiz olarak belirlenir.

İki süreç haberleşmek istediklerinde, kendi TCP'leri önce bir bağlantı oluşturmalıdır (her iki uçta da durum bilgileri ilklendirilmelidir). Haberleşme tamamlandığında, özkaynakları başkalarının kullanımı için serbest bırakmak için bağlantı sonlandırıldıktan sonra kapatılır.

Bağlantılar güvenilir olmayan makinalar arasında ve güvenilir olmayan Genel Ağ iletişim sistemi üzerinden yapılmak zorunda olunduğu için, bağlantıların hatalı ilklendirilmesinden kaçınmak için saate dayalı sıra numaraları ile bir uzlaşma mekanizması kulanılır.

Öncelik ve Güvenlik:

TCP kullanıcıları iletişimlerinin önceliğini ve güvenliğini belirleyebilir. Gerekli önlemler bu özellikler gerekli olmadığında kullanılan öntanımlı değerler için alınır.

2. FELSEFE

2.1. Ağlararası Sistemin Elemanları

Ağlararası ortam, ağgeçitleri üzerinden birbirlerine bağlanan ağlara bağlı konaklardan (ağların ağlararası ortama bağlanmasını sağlayan bilgisayarlardan) oluşur. Burada bahsedilen ağlar yerel ağlar (örn, ETHERNET) olabileceği gibi geniş ağlar da (örn, ARPANET) olabilir, ancak bunlar paket anahtarlamalı teknolojiyi kullanıyor olmalıdırlar. İletileri üreten de tüketen de süreçlerdir. Konaklar, ağgeçitleri ve ağlar çeşitli seviyelerdeki protokollerle portlar arasında mantıksal bağlantılardan iki yönlü veri akışı sağlayan süreçlerarası iletişim sistemini desteklerler.

Paket deyince burada genel olarak bir konakla ağ arasında bir hareketlik veriden bahsetmiş oluruz. Bir ağ içinde değiştokuş edilen veri bloklarının biçimi ise genellikle bizi ilgilendirmeyecektir.

Konaklar ağa bağlı bilgisayarlar olup haberleşme ağları bakımından ise paketlerin kaynakları ve hedefleridir. Süreçler (oldukça bilinen tanımı gereğince çalışmakta olan bir program olarak süreçler) ise konak bilgisayarlardaki etkin elemanlar olarak görülürler. Hatta uçbirimler, dosyalar veya G/Ç aygıtlarının bile bir diğeri ile süreçleri kullanarak haberleştikleri görülür. Böylece, tüm iletişim süreçler arasındaki haberleşme gibi görünür.

Bir sürecin kendisiyle başka bir süreç (veya süreçler) arasındaki çeşitli iletişim akımları arasında ayrım yapması gerekebileceğinden biz her sürecin bir miktar porta sahip olabileceğini ve diğer süreçlerin portlarıyla bunlar üzerinden haberleşeceğini farzediyoruz.

2.2. İşlem Modeli

Süreçler veriyi TCP'ye argüman olarak veri tamponlarını vererek yaptıkları çağrıyla aktarırlar. TCP bu tamponlardaki veriyi veribölütleri halinde paketler ve her veribölütünü hedef TCP'ye aktarmak üzere Genel Ağ modülünü çağırır. Alıcı TCP bir veribölütünden aldığı veriyi bir kullanıcı tamponuna yerleştirilir ve alıcı kullanıcıyı bilgilendirir. TCP'ler veri aktarımının düzgün sırada yapıldığından emin olunmasını sağlamakta kullanılmak üzere veribölütlerinde denetim bilgisi bulundururlar.

Genel Ağ iletişim modeli, yerel ağa bir arayüz sağlayan her TCP ile ilişkili birer Genel Ağ modülünden oluşur. Bu Genel Ağ modülü TCP veribölütlerini Genel Ağ verikatarları içinde paketler ve bu verikatarlarını hedef Genel

Ağ modülüne veya ara ağgeçitlerine yönlendirir. Verikatarını yerel ağ üzerinden aktarmak için verikatarı bir yerel ağ paketine gömülür.

Paket anahtarları yerel paketin hedef Genel Ağ modülüne teslimatını gerçekleştirmek için paketleme, parçalama ve diğer işlemleri uygulayabilir.

Ağlar arasındaki bir ağgeçidinde Genel Ağ verikatarı yerel paketten çıkarılır ve seyahatine hangi ağda devam edeceğini saptamak için incelenir. Genel Ağ verikatarı seyahatine devam edeceği ağa uygun bir yerel paket olarak yeniden sarmalanır ve sonraki ağgeçidine veya son hedefine yollanır.

Sonraki ağa aktarım için gerekliyse Genel Ağ verikatarı ağgeçidinde daha küçük Genel Ağ veridilimlerine parçalanabilir. Ağgeçidi bunu yapmak için her biri bir veridilimi taşıyan bir Genel Ağ verikatarı takımı üretir. Veridilimleri sonraki ağ geçitlerinde daha küçük veridilimlerine parçalanabilir. Genel Ağ veridilimlerinin biçimi hedef Genel Ağ modülünde tekrar Genel Ağ verikatarı oluşturmak üzere birleştirilebilecek şekilde tasarlanmıştır.

Ara hedeflerdeki Genel Ağ modülleri verikatarından bilgi dilimini çıkardıktan sonra (gerekliyse verikatarını yeniden oluşturduktan sonra) onu sonraki hedef TCP'ye aktarır.

Bu basit işlem modeli birçok ayrıntıyı gizler. Önemli bir özellik hizmet türüdür ve ağgeçidine (veya Genel Ağ modülüne) sonraki ağa geçirmekte kullanılacak hizmet parametrelerinin seçimine rehberlik edecek bilgiyi sağlar. Bu hizmet türü bilgileri içinde verikatarının öncelik bilgisi bulunur. Verikatarları ayrıca çok seviyeli güvenlikli ortamlarda çalışan konak ve ağgeçitlerinin güvenlik kaygılarıyla verikatarlarını bütünün içinden gerektiği gibi ayırmasını mümkün kılmak için güvenlik bilgisi taşıyabilir.

2.3. Konak Ortamı

TCP'nin, bir işletim sisteminde bir modül olduğu varsayılır. Kullanıcılar TCP'ye daha çok bir dosya sistemine erişir gibi erişirler. TCP işletim sisteminin diğer işlevleri üzerinden çağrılabilir; örneğin veri yapılarını yönetmek için. Asıl ağ arayüzünün bir aygıt sürücüsünün modülü tarafından denetlendiği varsayılır. TCP ağ aygıtı sürücüsünü doğrudan çağırmaz, çağrıyı sürücüye yöneltebilen Genel Ağ verikatarı protokol modülünü çağırır.

TCP mekanizmaları TCP'nin kullanıcı seviyesinde bir işlemcide gerçeklenimine engel olmaz. Yine de, böyle bir gerçeklenimde, konakla kullanıcı seviyesi arasındaki bir protokolle bu belgede açıklanan TCP kullanıcısı türü arayüzü destekleyecek işlevsellik sağlanmalıdır.

2.4. Arayüzler

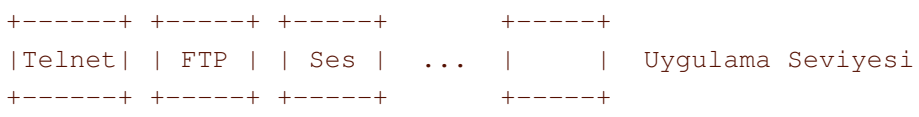
TCP/kullanıcı arayüzü, kullanıcı tarafından TCP üstünde bir bağlantının **AÇ**ılması/**KAPAT**ılması, verinin **GÖNDER**ilmesi/**AL**ınması veya bir bağlantı hakkında **DURUM** bilgisi sağlanması gibi çağrıların yapılması için tüm gereksinimleri sağlar. Bu çağrılar kullanıcı uygulamalarının işletim sistemine yaptığı diğer çağrılar gibi işlem görür; bir dosyanın açılması, okunması ve kapatılması gibi.

TCP/Genel Ağ arayüzü ise Genel Ağ sisteminin herhangi bir yerindeki konaklarda TCP modüllerine adreslenmiş verikatarlarının alınması ve gönderilmesi gibi çağrıların yapılması için tüm gereksinimleri sağlar. Bu çağrılar, adres, hizmet türü, öncelik, güvenlik ve diğer denetim bilgilerinin aktarılmasını sağlayacak parametrelere sahiptir.

2.5. Diğer Protokollerle İlişkiler

Aşağıdaki şekilde TCP'nin diğer protokoller arasındaki yeri gösterilmiştir:

Şekil 2. Protokol İlişkileri



Protokollerin Birbirleriyle İlişkileri

TCP'nin daha yüksek seviyeli protokolleri verimli bir şekilde desteklemeye muktedir olması beklenir. ARPANET Telnet veya AUTODIN II THP gibi yüksek düzeyli protokollere arayüzlük yapmak TCP için zor olmamalıdır.

2.6. Güvenilir İletişim

Bir TCP bağlantısı üzerinden gönderilen bir veri akımı hedefe güvenilir ve düzenli bir şekilde teslim edilir.

Aktarım, sıra numaraları ve alındılar kullanılarak güvenilir kılınır. Kavramsal olarak verinin her sekizlisine bir sıra numarası atanır. Bir veribölütündeki verinin ilk sekizlisinin sıra numarasına veribölütü sıra numarası denir ve veribölütü ile birlikte aktarılır. Veribölütleri ayrıca ters yöndeki aktarımın beklenen sonraki veri sekizlisinin sıra numarası olan bir alındı numarası taşır. TCP veri içeren bir veribölütünü aktardığında, bir kopyasını da yeniden aktarım kuyruğuna koyar ve zamanlayıcıyı başlatır; bu veri için bir alındı alındığında kopya veribölütü kuyruktan silinir. Zamanlayıcı sonlandığında hala alındı gelmemişse veribölütü yeniden aktarılır.

Bir alındının alınmış olması verinin son kullanıcıya teslim edildiğini garanti etmez, sadece alıcı TCP sorumluluğunun gereğini yapmıştır.

TCP'ler arasında veri akışını yönetmek için bir akış denetim mekanizması kurulmuştur. Alıcı TCP gönderici TCP'ye bir "pencere" raporlar. Bu pencere alındı numarasından başlanarak alıcı TCP'nin almaya hazır olduğu sekizliklerin sayısını belirtir.

2.7. Bağlantının Kurulması ve Temizliği

Bir TCP'nin işleme sokabileceği bağımsız veri akımlarını kimliklendirmek için TCP bir port tanıtı sağlar. Port tanıtları her TCP tarafından bağımsız olarak seçildiğinden eşsiz olmayabilirler. Her TCP'ye eşsiz tanıtlar sağlamak, yani birbirlerine bağlı ağlar üzerinde eşsiz olacak bir soket oluşturmak için Genel Ağ adresi ile port tanıtını birleştiririz.

Bir bağlantı, uçlarındaki soketlerin oluşturduğu çift ile belirtilir. Bir yerel soket çok sayıda yabancı soketle bağlantı kurabilir. Bir bağlantı veri taşımak için her iki yönde de kullanılabildiğinden "çift yönlü"dür.

TCP'ler hangi portun hangi süreçle ilişklendirileceğini belirlemekte özgürdürler çünkü onları kendileri seçerler. Bununla birlikte bazı gerçeklenimler bazı kabullerin yapılmasını gerektirir. TCP'nin bazı bakımlardan daima belli süreçleri belli soketlerle ilişkilendirdiğini bilmek gerekir. Biz, süreçlerin kendi portları olabileceğini ve bu süreçlerin sadece kendi soketleri üzerinden bağlantıya geçebileceğini, tasavvur ediyoruz. (Sahipliğin gerçeklenmesi yerel bir konudur ama bir port isteği yapacak bir kullanıcı komutu veya örneğin, belli bir süreci bir port isminin yüksek seviyeli bitleri ile ilişkilendirerek, belli bir sürece belli bir grup portu tahsis edecek bir yöntem tasarlayabiliriz.)

Bir bağlantı, argüman olarak bir yerel bir de yabancı port vererek bir **AÇ** çağrısıyda belirtilir. **AÇ** çağrısı geriye, kullanıcının sonraki çağrılar için atıfta bulunabileceği TCP tarafından sağlanmış bir (kısa) yerel bağlantı ismi döndürür. Bir bağlantı ile ilgili olarak hatırlanması gereken çeşitli şeyler vardır. Bu bilgiyi saklamak için Aktarım Denetim Bloğu (ADB) adını verebileceğimiz bir veri yapısı tasarlarız. Bir gerçeklenim stratejisi olarak yerel bağlantı ismi bu bağlantının ADB'sine bir gösterici olurdu. **AÇ** çağrısında ayrıca bağlantının doğrudan mı kurulacağı yoksa edilgen olarak mı bekleneceği belirtilir.

Edilgen bir **AÇ** isteği, sürecin bir bağlantı başlatmaktan ziyade gelen bağlantı isteklerini kabul etmek istediği anlamına gelir. Çoğunlukla edilgen bir **AÇ** isteği belirten bir süreç herhangi bir çağrıcıdan gelen bağlantı isteğini kabul edecektir. Bu durumda herşeyi sıfır bir yabancı soket, belirsiz bir soketi belirtmekte kullanılır. Belirsiz yabancı soketlere sadece edilgen **AÇ** çağrılarında izin verilir.

Bilinmeyen başka süreçlere hizmet sağlamak isteyen bir hizmet süreci belirsiz bir yabancı soketle edilgen bir **AÇ** çağrısı yapar. Bundan sonra bir süreç bu yerel sokete bağlantı isteğinde bulunmuş gibi yapılır. Eğer bu yerel soket bu hizmetle ilişkili olduğu bilinen bir soketse, bağlantı kurulur.

Ne oldukları bilinen soketler bir standart hizmeti öncelikle bir soket adresi ile ilişkilendiren uzlaşımsal bir mekanizmadır. Örneğin, "Telnet Sunucusu" süreci kalıcı olarak belli bir sokete atanmış ve diğer soketler de dosya aktarımı, uzak iş girişi, metin üreteci, yansıtıcı ve giriş noktası (son üçü deneme amaçlıdır) süreçleri için ayrılmıştır. Bir soket adresi, her yeniden oluşturuluşunda belli bir soket döndüren bir "Ara–Bul" hizmetine erişim için ayrılmış olabilirdi. Bildik soketler kavramı TCP belirtiminin bir parçasıdır, fakat soketlerin hizmetlere atanması bu belirtimin dışındadır (bkz, [*4*]).

Süreçler edilgen **AÇ** çağrıları yapabilir ve başka süreçlerin doğrudan **AÇ** çağrılarının bunlarla eşleşmesini beklerler; bağlantılar kurulduğunda TCP tarafından bilgilendirilirler. Birbirlerine aynı anda doğrudan **AÇ** çağrısı yapan iki süreç doğru biçimde bağlanacaktır. Bu esneklik bileşenleri birbirlerine göre eşzamansız hareket eden dağıtık hesaplamanın desteklenmesi bakımından yaşamsal önemdedir.

Yerel edilgen **AÇ** çağrıları ile yabancı doğrudan **AÇ** çağrılarındaki soketleri eşleştirmede başlıca iki durum sözkonusudur. İlk durumda, yerel edilgen **AÇ** çağrıları yabancı soketi tamamen belirtirler. Bu durumda eşleşme kesin olmalıdır. İkinci durumda, yerel edilgen **AÇ** çağrıları yabancı soketi belirsiz bırakırlar. Bu durumda, yerel soketler eşleştiği sürece herhangi bir yabancı soket kabul edilebilirdir. Diğer olasılıklar kısmen sınırlanmış eşleşmeleri kapsar.

Aynı yerel soketi kullanan edilgen **AÇ** çağrılarından bazıları (ADB'lerde kayıtlı olarak) beklemedeyse, belirsiz yabancı soketli bir ADB seçilmeden önce, bir yabancı doğrudan **AÇ** çağrısı, bir yerel doğrudan **AÇ** çağrısındaki belirli yabancı soketli bir ADB ile, eğer böyle bir ADB mevcutsa, (belirsiz yabancı soketli bekleyenleri sonraya bırakarak) eşleşecektir.

Bağlantıları oluşturan yordamlar eşzamanlama denetim bayrağından (`EŞZ`) yararlanırlar ve üç iletilik bir değiştokuşa katılırlar. Bu değiştokuş üçlü uzlaşı [*3*] diye adlandırılmıştır.

Bir bağlantı bir kullanıcı **AÇ** çağrısıyla oluşturulan ve beklemekte olan bir ADB ile `EŞZ` içeren bir veribölütünün geliş randevusu tarafından ilklendirilir. Yerel ve yabancı soketlerin eşleşmesi bir bağlantı ilklendirildiği anda sağlanmış olur. Sıra numaraları da her iki yönde eşzamanlandığı anda bağlantı "kurulmuş" olur.

Bir bağlantı temizliği de veribölütlerinin değiştokuşunu ve bu durumda `SON` denetim bayrağının taşınmasını gerektirir.

2.8. Veri İletişimi

Bir bağlantıdan akan veriyi bir sekizli akımı olarak düşünebiliriz. Gönderen taraf yaptığı bir **GÖNDER** çağrısındaki verinin beklemeksizin alıcı tarafa gidip gitmeyeceği `GİT`sin bayrağını kullanarak belirtir.

Gönderen TCP'nin gönderen taraftan veriyi toplamasına izin verilmiş ve gitsin işlevi algılanıncaya kadar verinin veribölütleri halinde gönderilmesi kararı da ona bırakılmıştır, ancak gitsin işlevini algıladığı anda henüz gönderilmemiş verinin tümünü göndermek zorundadır. Alıcı TCP GİTsin bayrağını gördüğünde, veriyi alıcı sürece aktarmadan önce gönderen TCP'den artık veri gelmesini beklememelidir.

Veribölütü sınırları ile gitsin işlevi arasında bir ilişkinin varlığı gerekli değildir. Herhangi bir veribölütündeki veri tek bir **GÖNDER** çağrısının sonucu olabileceği gibi çok sayıda **GÖNDER** çağrısının kısmen veya tamamının sonucu olabilir.

GİTsin bayrağının ve gitsin işlevinin amacı verinin beklemeksizin gönderici taraftan alıcı tarafa gitmesini sağlamaktır. Bir kayıt hizmeti sağlamaz.

TCP/kullanıcı arayüzünden geçen verinin tamponlarının kullanımı ile gitsin işlevi arasında bir uyum vardır. Bir GİTsin bayrağı her zaman alıcı tarafın tamponuna yerleştirilecek veri ile ilişkilidir, tampon dolmamış bile olsa tampon işlenmek üzere kullanıcıya döndürülür. Eğer bir GİTsin görünmeden önce gelen veri tampondan taşacak kadar fazla ise tampon doldukça veri tampon boyutluk birimler halinde kullanıcıya aktarılır.

TCP ayrıca verilerin alıcısı ile iletişime geçerek veri akımı içindeki bir noktada alıcının acil veriyi okumakta olduğunu bildirmekte kullanılır. TCP kullanıcıya bekleyen acil bir veri olduğunu haber verdikten sonra alıcının ne yapacağını tanımlaya çalışmaz, ama bu noktadaki genel eğilim alıcı sürecin acil veriyi hemen işleme alacağıdır.

2.9. Öncelik ve Güvenlik

TCP, her bağlantının önceliğini ve güvenliğini sağlamak için Genel Ağ protokolünün hizmet türü alanı ile güvenlik seçeneğini TCP kullanıcısına kullanılır kılar. Tüm TCP modüllerinin çok seviyeli bir güvenlik ortamında işlevsel olması gerekli değildir; bir kısmı sadece sınıflanmamış kullanımla sınırlıyken bir kısmı da sadece tek bir güvenlik seviyesinde ve bölmesinde çalışabilir. Bu nedenle, bazı TCP gerçeklenimleri ve kullanıcı hizmetleri çok seviyeli güvenlik durumunun bir alt kümesiyle sınırlı olabilir.

Çok seviyeli bir güvenlik ortamında işleyen TCP modülleri giden veribölütlerini güvenlik, bölüm ve öncelik bakımından düzgün biçimde imlemelidir. Bu tür TCP modülleri kullanıcılarına veya Telnet ve THP gibi daha üst düzey protokollere ayrıca, bağlantılarda istenen güvenlik seviyesini, bölmesini ve önceliği belirtmelerine imkan veren bir arayüz sağlamalıdır.

2.10. Güçlülük İlkesi

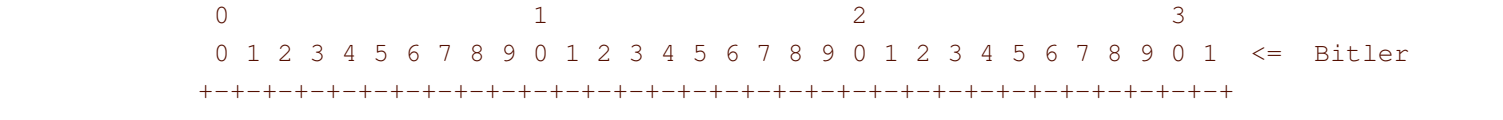
TCP gerçeklenimleri genel güçlülük ilkesini izleyecektir: yapılacaklar konusunda tutucu, başkalarından kabul edilecekler konusunda özgürlükçü olmak.

3. İşlevsel Belirtim

3.1. Başlık Biçimi

TCP veribölütleri Genel Ağ verikatarları olarak gönderilir. Genel Ağ Protokolünün başlığı [2] kaynak ve hedef konakların adreslerinin de içinde bulunduğu çeşitli bilgi alanları içerir. Genel Ağ Protokolünün başlığını TCP'ye özgü bilgi sağlayan TCP başlığı izler. Bu kısım TCP'ye ek olarak başka konak seviyesi protokollerin varlığına da izin verir.

Şekil 3. TCP Başlık Biçimi



TCP Başlık Biçimi

Kaynak Port: 16 bit
: Kaynak port numarası

Hedef Port: 16 bit
: Hedef port numarası

Sıra Numarası: 32 bit
: Bu veribölütündeki ilk veri sekizlisinin sıra numarası (EŞZ varlığı hariç). Eğer EŞZ varsa sıra numarası ilk sıra numarasıdır (İSN) ve ilk veri sekizlisi de İSN+1'dir.

Alındı Numarası: 32 bit
: ALN denetim biti etkinse bu alan veribölütü göndericisinin alacağı umulan sonraki sıra numarasını içerir. Bir bağlantı kurulduğu anda bu daima gönderilir.

Veri Başlangıcı: 4 bit
: TCP Başlığındaki 32 bitlik sözcüklerin sayısı. Bu verinin başladığı yeri belirtir. TCP başlığı (bir seçenek içerse bile) 32 bitlik bir sayıdır.

Yedek: 6 bit
: İlleride kullanmak üzere yedek. Sıfır olmalı.

Denetim bitleri: 6 bit (soldan sağa):
: ACL: Aciliyet Göstergesi alanı önemli
: ALN: Alındı alanı önemli
: GİT: Gitsin işlevi
: BŞT: Bağlantıyı baştan al
: EŞZ: Sıra numaralarını eşzamanla
: SON: Göndericide başka veri yok

Pencere: 16 bit
: veribölütü göndericisinin kabul etmeye hazır olduğunu belirttiği alındı alanı ile başlayan veri sekizliklerinin sayısı.

Sağlama Özeti: 16 bit

Başlık ve metindeki 16 bitlik sözcüklerin bire tümleyen toplamının bire tümleyenini içeren 16 bitlik sağlama özeti alanı. Eğer bir veribölütü sağlama toplamı alınmak üzere başlık ve metin olarak tek sayıda sekizli içeriyorsa, sağlama özetinin amaçlarına uygun olarak 16 bitlik biçimi sağlamak için son sekizlinin sağındaki alan sıfırlarla doldurulur. Dolgu olarak kullanılan sekizli veribölütünün parçası olarak aktarılmaz. Sağlama özeti hesaplanırken sağlama özeti alanının kendisi sıfırlarla doldurulur.

Ayrıca sağlama özeti kavramsal olarak TCP başlığına önek olarak 96 bitlik sözde başlık bilgisini de içerir. Bu sözde başlık bilgisi Kaynak Adres, Hedef Adres, Protokol ve TCP uzunluğunu içerir. Bu, yanlış yollanmış veribölütlerine karşı TCP koruması sağlar. Bu bilgi Genel Ağ Protokolü tarafından taşınır ve TCP tarafından IP'ye yapılan çağrıların sonuçlarında veya argümanlarında TCP/ağ arayüzüne aktarılır.

```
+--------+--------+--------+--------+
|           Kaynak Adres             |
+--------+--------+--------+--------+
|            Hedef Adres             |
+--------+--------+--------+--------+
|  sıfır | SBB Uz |  TCP Uzunluğu    |
+--------+--------+--------+--------+
```

TCP Uzunluğu, TCP başlık uzunluğu ile sekizliler cinsinden veri uzunluğunun toplamıdır (Bu miktar bilgisi doğrudan aktarılmaz, hesaplanır) ve sözde başlığın 12 sekizlisi sayılmaz.

Aciliyet Göstergesi: 16 bit

Bu alan, bu veribölütündeki sıra numarasından pozitif mesafe olarak aciliyet göstericisinin o anki değerini nakleder. Aciliyet göstericisi acil verinin başındaki sekizlinin sıra numarasını gösterir. Bu alan sadece ACL denetim biti etkin olan veribölütlerinde yorumlanır.

Seçenekler: değişken

Seçenekler TCP başlığının sonunda 8 bitin katları olarak yer işgal edebilirler. Tüm seçenekler sağlama özetinde içerilirler. Bir seçenek herhangi bir sekizlinin sınırından başlayabilir. Bir seçeneğin biçimi ile ilgili iki durum vardır:

- 1. durum: Seçenek çeşidini içeren tek bir sekizli.
- 2. durum: Seçenek çeşidi için bir, seçenek uzunluğu için bir sekizli ve asıl seçenek verisi sekizlileri.

Seçenek uzunluğu olarak, çeşit bir ve uzunluk iki saydıktan sonra asıl veri sekizlileri sayılır.

Seçenek listesinin veri başlangıcı alanının gerektirdiğinden daha kısa olabileceği gözden uzak tutulmamalıdır. Başlık içeriği, "Seçenek Listesi Sonu" seçeneğinden sonra başlık dolgu karakteri (örn, sıfır) ile doldurulmalıdır.

Bir TCP tüm seçenekleri gerçeklemelidir.

Şimdilik tanımlı seçenekler şunlardır (çeşit sekizli cinsinden belirtilir):

Çeşit	Uzunluk	Anlamı
0	-	Seçenek Listesi Sonu.
1	-	İşlem Yok.
2	4	Azami Veribölütü Boyu.

Özel Seçenek Tanımları

Seçenek Listesi Sonu

```
+--------+
|00000000|
+--------+
  Çeşit=0
```

Bu seçenek kodu Seçenek listesinin sonunu belirtir. Bu, Veri Başlangıcı alanı ile ilgili TCP başlığının sonu ile çakışık olmayabilir. Bu, her seçeneğin sonunda değil, bütün seçeneklerin sonunda kullanılır ve sadece TCP başlığının sonu ile seçeneklerin sonu çakışık değilse kullanımına ihtiyaç duyulur.

İşlem Yok

```
+--------+
|00000001|
+--------+
  Çeşit=1
```

Bu seçenek kodu seçenekler arasında kullanılabilir, örneğin sonraki seçeneğin sözcük sınırına hizalanmasında. Göndericilerin bu seçeneği kullanacağının hiç bir garantisi yoktur, bu yüzden alıcılar bir sözcük sınırından başlamasa bile tüm seçenekleri işleme sokabilmeye hazır olmalıdır.

Azami Veribölütü Boyu

```
+--------+--------+---------+--------+
|00000010|00000100|    azm vb boyu    |
+--------+--------+---------+--------+
  Çeşit=2   Uzunluk=4
```

Azami Veribölütü Boyu Seçenek Verisi: 16 bit

Bu seçenek mevcutsa, seçenekle bu veribölütünü gönderen TCP'deki azami alım veribölütü boyu nakledilir. Bu alan sadece ilk bağlantı isteğinde (örn, `EŞZ` denetim bitli veribölütlerinde) gönderilmelidir. Bu seçenek kullanılmamışsa her veribölütü boyuna izin verilir.

Dolgu: değişken

TCP başlığının 32 bitlik olmasını sağlamak için dolgu yapılır. Doldurma işleminde sıfırlar kullanılır.

3.2. Terminoloji

TCP işlemlerine çok fazla dalmadan önce terminoloji üzerinde ayrıntılı olarak durmak istiyoruz. Bir TCP bağlantısının sürdürülmesi çeşitli değişkenlerin hatırlanmasını gerektirir. Bu değişkenlerin saklanacağı yer olarak Aktarım Denetim Bloğu (ADB) adını verdiğimiz bir bağlantı kaydı tasarladık. Yerel ve uzak soket numaralarını, bağlantı önceliği ve güvenliği, kullanıcıların gönderim ve alım tamponların göstericileri yeniden aktarım kuyruğunun göstericisi ve o anki veribölütü, ADB'de saklanan değişkenler arasında sayılabilir. Ek olarak gönderim ve alım sıra numaraları ile ilgili çeşitli değişkenler de ADB'de saklanır.

Gönderi Sıra Numaraları Değişkenleri

```
GÖN.OLM - gönderi alınmadı
GÖN.SNR - sonraki gönderi
GÖN.PEN - gönderi penceresi
GÖN.ACL - gönderi aciliyet göstericisi
GÖN.WL1 - son pencere güncellemesinde kullanılan veribölütü sıra numarası
GÖN.WL2 - son pencere güncellemesinde kullanılan veribölütü alındı numarası
İGS     - ilk gönderi sıra numarası
```

Alım Sıra Numaraları Değişkenleri

```
ALM.SNR - sonraki alım
ALM.PEN - alım penceresi
ALM.UP  - alım aciliyet göstericisi
İAS     - ilk alım sıra numarası
```

Bu değişkenlerden bazılarının sıra değişkenleri uzayıyla ilişkilendirlmesinde aşağıdaki şeklin yardımı olabilir.

Şekil 4. Gönderi Sıra Numaraları Uzayı

```
          1         2          3          4
     ----------|----------|----------|----------
            GÖN.OLM    GÖN.SNR    GÖN.OLM
                                  +GÖN.PEN

  1 - alındı'lanmış eski sıra numaraları
  2 - alındı'lanmamış sıra numaraları
  3 - yeni veri aktarımı için izin verilen sıra numaraları
  4 - henüz izin verilmemiş gelecekteki sıra numaraları

           Gönderi Sıra Numaraları Uzayı
```

Gönderi penceresi *Gönderi Sıra Numaraları Uzayı* (sayfa: 15)'teki sıra numaraları uzayının 3 ile etiketlenmiş bölümüdür.

Şekil 5. Alım Sıra Numaraları Uzayı

```
          1         2          3
     ----------|----------|----------
            ALM.SNR    ALM.SNR
                      +ALM.PEN

  1 - alındı'lanmış eski sıra numaraları
  2 - yeni alım için izin verilen sıra numaraları
  3 - henüz izin verilmemiş gelecekteki sıra numaraları

          Alım Sıra Numaraları Uzayı
```

Alım penceresi *Alım Sıra Numaraları Uzayı* (sayfa: 16)'teki sıra numaraları uzayının 2 ile etiketlenmiş bölümüdür.

Ayrıca değerlerini o anki veribölütündeki alanlardan alan ve incelememizde sıkça kullanılacak olan birtakım değişkerler daha vardır.

O anki Veribölütü Değişkenleri

```
VBL.SIRA - veribölütü sıra numarası
VBL.ALN  - veribölütü alındı numarası
VBL.UZN  - veribölütü uzunluğu
VBL.PEN  - veribölütü penceresi
VBL.ACL  - veribölütü aciliyet göstericisi
VBL.PRC  - veribölütü öncelik değeri
```

Bir bağlantı yaşamı boyunca bir seri durumdan geçer. Bu durumlar: `DİNLE`, `EŞZ-GÖNDER`, `EŞZ-ALINDI`, `KURULU`, `SON-BEKLE-1`, `SON-BEKLE-2`, `KAPAT-BEKLE`, `KAPANIŞ`, `SON-ALN`, `ZMN-BEKLE` ve kurgusal durum `KAPALI`. `KAPALI` kurgusaldır, çünkü hiç ADB'nin olmadığı zamanı ve dolayısıyla bağlantı yokluğunu ifade eder. Durumların kısaca anlamları:

`DİNLE`
: Bir porttan bir TCP bağlantısının beklendiğini gösterir.

`EŞZ-GÖNDER`
: Bir bağlantı isteği gönderildikten sonra bununla eşleşen bir bağlantı isteğinin beklendiğini gösterir.

`EŞZ-ALINDI`
: Bir bağlantı isteği gönderilip alındıktan sonra onaylayıcı bağlantı isteği alındısının beklendiğini gösterir.

KURULU
: Alınan verinin kullanıcıya teslim edilebileceği bir açık bağlantıyı gösterir. Bu bağlantının veri aktarımının yapıldığı normal fazıdır.

SON-BEKLE-1
: Uzak TCP'den bir bağlantı sonlandırma isteğinin ya da evvelce gönderilmiş bağlantı sonlandırma isteğinin alındısının beklendiğini gösterir.

SON-BEKLE-2
: Uzak TCP'den bir bağlantı sonlandırma isteğinin beklendiğini gösterir.

KAPAT-BEKLE
: Yerel kullanıcıdan bir bağlantı sonlandırma isteğinin beklendiğini gösterir.

KAPANIŞ
: Uzak TCP'den bir bağlantı sonlandırma isteği alındısının beklendiğini gösterir.

SON-ALN
: Uzak TCP'ye evvelce gönderilmiş (bir bağlantı sonlandırma isteği alındısı içeren) bağlantı sonlandırma isteğinin alındısının beklendiğini gösterir.

ZMN-BEKLE
: Uzak TCP'nin bağlantı sonlandırma isteğinin alındısını almasına kadar geçecek sürenin beklendiğini gösterir.

KAPALI
: Artık bir bağlantının olmadığını gösterir.

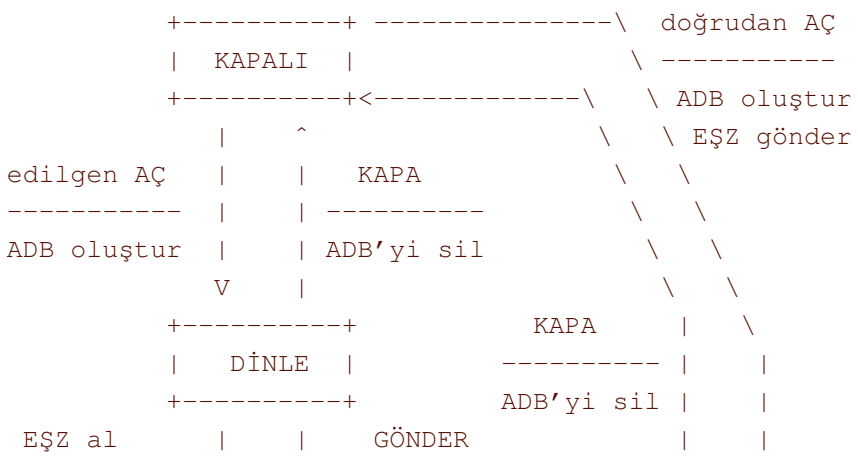
Bir TCP bağlantısı yanıtlarla bir durumdan diğerine geçilen bir süreçtir. Yanıtların sebepleri **AÇ**, **GÖNDER**, **AL**, **KAPAT**, **TERKET** ve **DURUM** kullanıcı çağrıları; özellikle EŞZ, ALN, BŞT ve SON bayraklarını içeren gelen veribölütleri; ve zamanaşımlarıdır.

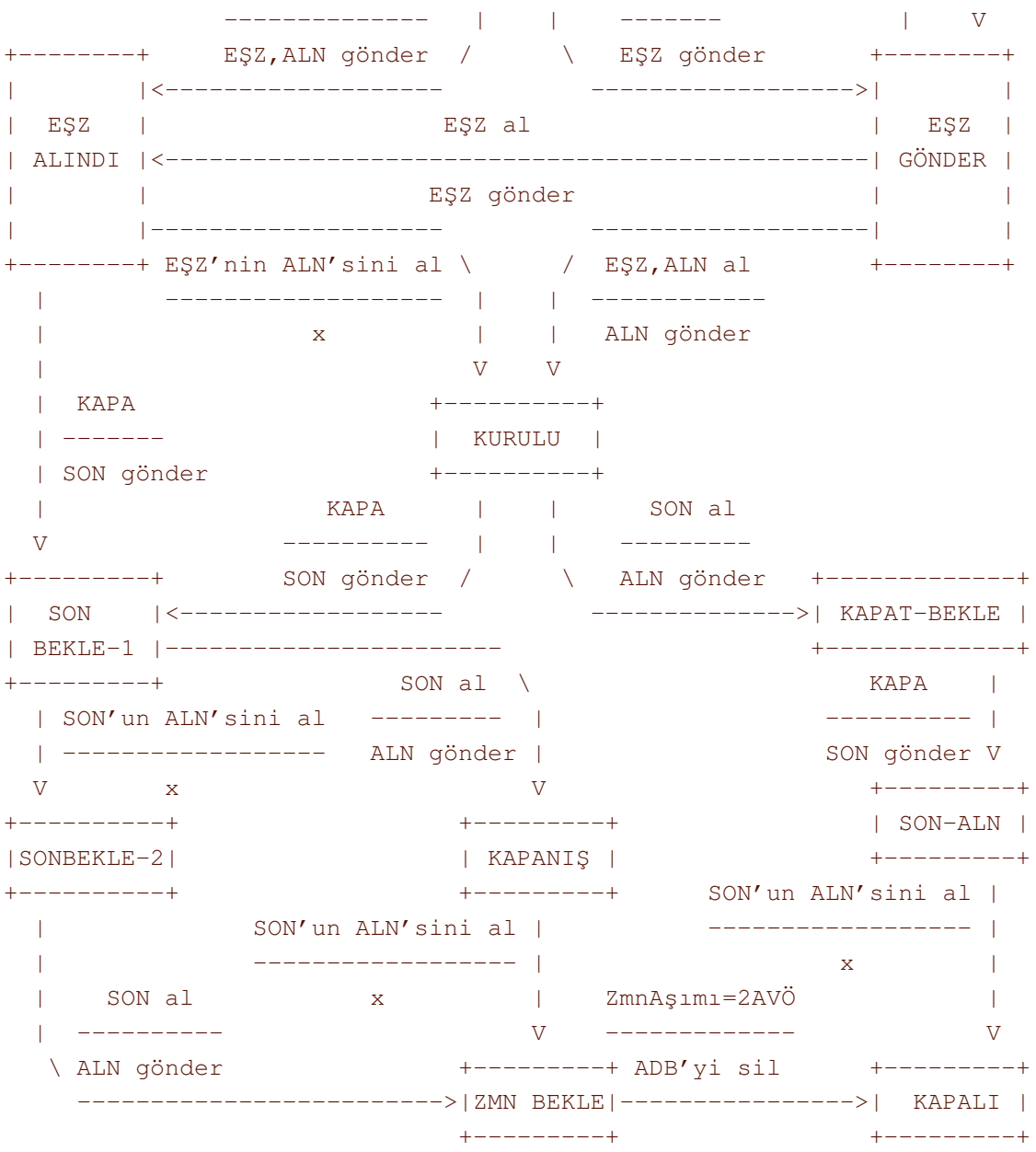
TCP Bağlantı Durumları Şeması (sayfa: 17)'daki durum şeması sadece sebepleri ve sonuçlarıyla durum değişikliklerini görselleştirir. Fakat durum değişikliklerine bağlı hata durumları ve eylemler dikkate alınmamıştır. Sonraki bölümde TCP'nin olaylara tepkisi daha ayrıntılı incelenecektir.

Dikkat

Bu şema sadece bir özettir ve belirtimin tamamı olarak ele alınmamalıdır.

Şekil 6. TCP Bağlantı Durumları Şeması





TCP Bağlantı Durumları Şeması

3.3. Sıra Numaraları

Tasarımdaki temel bir fikir bir TCP bağlantısından gönderilen her veri sekizlisinin bir sıra numarısının olmasıdır. Sekizliler sıralı olduklarından onlar için alındı gönderilebilir. Kullanılacak alındı mekanizması kümeleme yapan türdedir, böylece X numaralı alındı ile X numaralı sekizli hariç X'inci sekizliye kadar olan sekizlilerin alındığı belirtilebilir. Bu mekanizma, yeniden aktarım gerektiğinde düz mantıkla yinelenmenin saptanmasına da olanak sağlar. Bir veribölütündeki sekizliler, başlıktan hemen sonra gelen sekizli en düşük numarayı alacak şekilde numaralandırılırlar.

Gerçekte sıra numarası uzayının çok büyük olmasına rağmen sonlu olması gerektiğini unutmamak esastır. Bu uzay $2^{32} - 1$ birimliktir. Uzay sonlu olduğundan, sıra numaraları ile ilgili tüm işlemler 2^{32}'nin katlarında $2^{32} - 1$ birimlik kümeler (ve hatta aynı sınırlama ile küme kümeleri) halinde yapılmalıdır. Bu tür işlemlerde bazı cinlikler olabildiğinden yazılımı geliştirirken böyle değerlerle karşılaşılacağı dikkate alınmalıdır. "=<" sembolü "küçük veya eşittir" anlamına gelir.

TCP'nin uygulayacağı sıra numaraları kıyaslamalarından başlıcaları:

a. gönderilmiş ama henüz alınmamış bir sıra numarasını gösteren bir alındının saptanması.
b. Alındı'lanmış bir veribölütü tarafından işgal edilen tüm sıra numaralarının saptanması (örneğin, veribölütünün yeniden aktarım kuyruğundan silinmesi için).
c. Beklenen sıra numaralarını içeren bir gelen veribölütünün saptanması (örneğin, bu veribölütü alım penceresi ile "örtüşüyordur").

Gönderilen veriye verilen yanıtta TCP, alındıları alır. Alındıların işleme konulmasında aşağıdakilerle ilgili kıyaslamaların yapılması gerekir:

```
GÖN.OLM = en eski alındı'lanmamış sıra numarası

GÖN.SNR = gönderilecek sonraki sıra numarası

VBL.ALN = Alan TCP'deki alındı numarası
          (alan TCP'nin beklediği sonraki alındı numarası)

VBL.SIRA = bir veribölütünün ilk sıra numarası

VBL.UZN = veribölütündeki veri tarafından işgal edilen sekizlilerin sayısı
          (EŞZ ve SON sayarak)

VBL.SIRA+VBL.UZN-1 = bir veribölütündeki son sıra numarası
```

Yeni bir alındı numarası aşağıdaki eşitsizlikle ifade edilen değerlerden biridir:

```
GÖN.OLM < VBL.ALN =< GÖN.SNR
```

Yeniden aktarım kuyruğundaki bir veribölütü, eğer sıra numaralarının toplamı ya da uzunluğu gelen veribölütündeki alındı değerinden küçük veya ona eşitse, tamamen alındı'lanmış demektir.

Veri alındığında aşağıdaki kıyaslamaların yapılması gerekir:

```
ALM.SNR = gelen veribölütünde beklenen sonraki sıra numarası olup
          alım penceresinin alt veya sol kenarıdır

ALM.SNR+ALM.PEN-1 = gelen veribölütünde beklenen son sıra numarası olup
                    alım penceresinin üst veya sağ kenarıdır

VBL.SIRA = gelen veribölütü tarafından işgal edilen ilk sıra numarası

VBL.SIRA+VBL.UZN-1 = gelen veribölütü tarafından işgal edilen son sıra numarası
```

Bir veribölütünün geçerli sıra numarası uzayının bir kısmını işgal ettiğine karar verilebilmesi için şunlardan biri geçerli olmalıdır:

```
ALM.SNR =< VBL.SIRA < ALM.SNR+ALM.PEN
```

veya

```
ALM.SNR =< VBL.SIRA+VBL.UZN-1 < ALM.SNR+ALM.PEN
```

Veribölütünün başlangıcının pencere içine düşüp düşmediğini anlamak için bu sınamanın ilk parçası, sonunun pencere için düşüp düşmediğini anlamak için ise ikinci parçası kullanılır. Eğer veribölütü her iki sınamayı da aşarsa penceredeki veriyi içeriyor demektir.

Aslında durum bundan biraz daha karışıktır. Sıfır sayıda pencere ve sıfır uzunlukta veribölütü olabileceğini de gözönüne alarak bir gelen veribölütünün kabul edilebilirliği için dört durum söz konusudur:

```
Veribölütü    Alım      Sınama
 Uzunluğu   Penceresi
----------  ---------  ------------------------------------------

    0           0       VBL.SIRA = ALM.SNR

    0          >0       ALM.SNR =< VBL.SIRA < ALM.SNR+ALM.PEN

   >0           0       kabul edilebilir değil

   >0          >0       ALM.SNR =< VBL.SIRA < ALM.SNR+ALM.PEN
                     veya ALM.SNR =< VBL.SIRA+VBL.UZN-1 < ALM.SNR+ALM.PEN
```

Alım penceresi sıfır olduğunda ALN veribölütleri dışında hiçbir veribölütünün kabul edilmeyeceği dikkate alınmalıdır. Bu durumda, bir TCP için veriyi aktarırken ve ALN'leri alırken sıfır alım penceresi sağlanması olasıdır. Yine de, alım penceresi sıfır olduğunda bile, bir TCP tüm gelen veribölütlerindeki BŞT ve ACL alanlarını işleme sokmalıdır.

Numaralama şemasının yararlarını belli denetim bilgilerini korumakta da kullanabiliriz. Bunu sıra numaraları uzayına örtük olarak bazı denetim bayraklarını dahil ederek sağlayabiliriz, böylece bunlar yeniden aktarılabilir ve bir karışıklık olmaksızın alındılanabilir (yani bir ve yalnız bir denetim kopyası rol alacak şekilde). Denetim bilgisi veribölütü veri uzayında fiziksel olarak taşınmaz. Bu bakımdan, denetim bilgilerine sıra numaralarını örtük olarak atamak için bazı kurallar uydurmalıyız. EŞZ ve SON bayraklarından başka bu korumaya ihtiyaç duyan denetim yoktur ve bu denetimler sadece bağlantının açılışında ve kapanışında kullanılır. Sıra numaralarının amaçlarına uygun olarak, EŞZ'nin kendisi için oluştuğu veribölütünün ilk asıl veri sekizliğinden önce oluştuğu düşünülürken, SON'un kendisi için oluştuğu veribölütünün son asıl veri sekizliğinden sonra oluştuğu düşünülür. Veribölütü uzunluğu (VBL.UZN) hem veri hem de denetimin işgal ettiği sıra numarası uzayını kapsar, Bir EŞZ olduğunda, VBL.SIRA'nın değeri EŞZ'nin sıra numarasıdır.

3.3.1. İlk Sıra Numarasının Seçimi

Protokol bir bağlantı üzerinde hiçbir sınırlama olmaksızın tekrar tekrar kullanılır. Bir bağlantı bir soket çifti ile tanımlanır. Bir bağlantının yeni örneklerine bağlantının yeniden varolmuşu gibi başvurulur. Bu noktada sorun, TCP'nin bağlantının önceki yeniden varouşlarından kalan yinelenmiş veribölütlerini nasıl tanıyacağıdır. Bu sorun bağlantı çabucak açılıp kapatılırsa veya bağlantı bellek kaybı nedeniyle kopar ve yeniden kurulursa görünür hale gelir.

Karışıklıktan kaçınmak için bir bağlantının yeniden oluşturulmuşundaki veribölütlerinden, ağ üzerinde hala aynı sıra numaralarıyla mevcut olanlarının kullanımını engelleyeceğiz. Bir TCP kopup kullanımdaki sıra numaralarına ait tüm bilgi kaybolsa bile bunu sağlama almak isteriz. Yeni bağlantılar oluşturulduğunda, yeni bir 32 bitlik ilk sıra numarası (İSN) seçen bir İSN üreteci devreye sokulur. Üreteç, en düşük biti kabaca her 4 mikrosaniyede bir artan 32 bitlik (muhtemelen kurgusal) bir saate bağlıdır. Bu durumda, İSN yaklaşık her 4,55 saatte bir başa döner. Veribölütleri ağda Azami Veribölütü Ömrü'nden (AVÖ) daha fazla kalmayacağından ve AVÖ 4,55 saatten küçük olduğundan normal olarak İSN'nin eşsiz olacağını varsayarız.

Her bağlantı için bir gönderi sıra numarası bir de alım sıra numarası vardır. İlk gönderi sıra numarası (İGS) veriyi gönderen TCP tarafından seçilirken ilk alım sıra numarası (İAS) ise bağlantının kuruluş işlemleri sırasında öğrenilir.

Bir bağlantının kurulması ve ilklendirilmesi için iki TCP'nin birbirlerinin ilk sıra numaraları üzerinden

eşzamanlanmaları gerekir. Bu, "EŞZ" denetim bitini ve ilk sıra numarasını taşıyan veribölütlerini oluşturan bağlantı değiştokuşunda yapılır. Bir kısaltma olarak EŞZ biti taşıyan veribölütlerine "EŞZler" de denilmektedir. Dolayısıyla, çözüm, ilk sıra numarasının seçimi için uygun bir mekanizma ve İSN'leri değiştokuş etmek için biraz karışık bir uzlaşım mekanizması gerektirir.

Eşzamanlama iki tarafın birbirlerine kendi ilk sıra numaralarını göndermelerini ve bunların alındılarılarla onaylanmasını gerektirir.

1. A --> B EŞZ sıra numaram X
2. A <-- B ALN sıra numaranız X
3. A <-- B EŞZ sıra numaram Y
4. A --> B ALN sıra numaranız Y

2. ve 3. adım tek bir ileti olarak birleştirilebildiğinden buna *üçlü uzlaşım* adı verilir.

Üçlü uzlaşımın gerekli olmasının sebebi sıra numaralarının ağdaki genel bir saate göre berabere kalamamaları ve TCP'lerin İSN'leri farklı mekanizmalarla seçebilmeleridir. İlk EŞZ'nin alınmasıyla, bağlantıda kullanılan son sıra numarası hatırlanmadıkça (her zaman mümkün değildir), veribölütünün eskiden kalma olup olmadığını anlamak mümkün değildir ve dolayısıyla göndericiden bu EŞZ'yi doğrulaması istenmelidir. Üçlü uzlaşım ve saatli işlem şemasının yararları [*3*]'te açıklanmıştır.

3.3.2. Susmayı Bilmek

TCP'nin ağda eskiden kalmış bir veribölütünün yinelenmişinin sıra numarasını taşıyan bir veribölütü oluşturmadığından emin olmak için, bağlantının yenisinin başlatılması veya kullanımdaki sıra numarası belleğinin kaybından dolayı bir bozulmanın kurtarılması amacıyla bir sıra numarası atamadan önce, TCP'nin bir AVÖ boyunca sessiz kalması gerekir. Bu belirtim için AVÖ 2 dakika olarak alınmıştır. Bu bir mühendislik seçimidir ve tecrübeler başka bir değeri gerektiriyorsa değiştirilebilir. Bir nedenle TCP yeniden ilklendirildiğinde sıra numarası belleği hala kullanımdaysa hiç beklenmemesi gerekir; emin olunması gereken tek şey kullanılacak sıra numarasının son kullanılandan büyük olacağıdır.

3.3.2.1. TCP Sessizlik Süresi Kavramı

Bu belirtim bize gösterecek ki, Genel Ağ sisteminin bir parçası olan konakların etkin (kapalı olmayan) bağlantılar üzerinden aktarılan son sıra numaraları ile ilgili hiç bir bilgi kalmaksızın "çökme" leri durumunda, TCP'nin herhangi bir veribölütünü iletmesi en azından kabul edilen AVÖ değeri kadar gecikecektir. Aşağıdaki paragraflarda, bu belirtim için bir açıklama yapılmıştır. TCP gerçeklenimcileri bu "sessizlik süresi" sınırlamasını zorlayabilirler, fakat tek risk eski verilerin yeni olarak kabul edilmesine veya yeni veriyi Genel Ağ sistemindeki bazı alıcıların eskinin yinelenmişi olarak reddetmelerine sebep olmaktır.

TCP'ler veribölütlerinin biçimlendirilip kaynak konağın ağ çıkış kuyruğuna her atılışında sıra numarası uzayını tüketirler. TCP'deki sıralama algoritması ve yinelenmişin saptanması için güvenilen tek şey veribölütü alıcıya ulaştırılıp onay alınmadan ve veribölütünün Genel Ağ üzerinde bulunan bütün kopyaları yok edilmeden veribölütü verisinin sıra numarasının eşsizliğini sağlayan sıra numaralarının 2^{32} değerinin hepsi tüketilmeden döngünün yeniden başlatılmayacağıdır. Böyle bir önkabul olmaksızın, iki ayrı TCP veribölütüne aynı sıra numaralarının atanmış olması veya üst üste gelmesi gibi bir olasılıklar doğar ve bu da alıcı tarafta hangi verinin yeni hangisinin eski olduğu konusunda karmaşaya yol açar. Hatırlarsanız bir veribölütünün içinde veri sekizlileri olduğu kadar çok sayıda ardışık sıra numarası da bulunuyordu.

Normal şartlar altında, alındılanmış ilk sıra numarasından önce bir sıra numarasının tekrar kullanılmasını önlemek için TCP'ler kullanılacak sonraki sıra numarası ile bekleyen en eski alındının kaydını tutarlar. Bu tek başına sıra numarasının, kullanılmadan önce eski yinelenmiş verinin ağ üzerinden silinmiş oluşunu garantilemez, bu nedenle başıboş bir yinelenmişin, istenmeyen bir varış olasılığını düşürmek için sıra numarası uzayı

çok geniş tutulur. Sıra numarası uzayının 2^{32} sekizlisinin kullanımı 2 megabit/saniye hızında 4.5 saat sürer. AVÖ ağ üzerinde birkaç on saniyeyi aşamayacağından veri hızı onlarca megabit/saniye hızlara tırmansa bile, bu, öngörülebilir ağlar için yeterince iyi bir koruma sayılır. 100 megabit/saniye için döngü tamamlanma süresi 5.4 dakika olur ve bu hala makul bir koruma sağlar.

Eğer bir kaynak TCP son kullandığı bağlantıda sıra numaraları belleğine sahip değilse, TCP'deki yineleme saptama ve sıralama algoritması yine de bozulabilir. Örneğin, bir TCP tüm bağlantıları 0 sıra numarası ile başlatıyorsa, bir çökme veya yeniden başlatma sebebiyle, bu TCP önceki bağlantıyı (muhtemelen yarı açık bağlantı çözünürlüğü sonrası) yeniden şekillendirebilir ve aynı bağlantının önceki bir varoluşunda kullanılmış paketlerden ağda halen varlığını sürdürenlerle çakışan veya onlarla aynı sıra numaralarına sahip paketler gönderebilir. Belli bir bağlantıda kullanılmış sıra numaraları ile ilgili bilginin yokluğunda, TCP belirtimi veribölütlerinin bağlantıya gönderilmeden önce, bağlantının önceki varoluşlarına ait sistemdeki veribölütlerinin kaldırılması için gereken süreyi sağlamak için, AVÖ saniyelik bir kaynak gecikmesi uygulanmasını önerir.

Tam gün zamanı hatırlayan ve ilk sıra numarasının seçiminde bunu kullanan konaklar bile bu sorundan bağışık değillerdir. (Örneğin, her yeni bağlantının yeniden varoluşunda ilk sıra numarasını seçerken.)

Varsayalım ki, örneğin, S sıra numarası ile başlayan bir bağlantı açılmış olsun. Bu bağlantının çok kullanılmadan eninde sonunda ilk sıra numarası işlevinin (İSN(t)) S1 diye, bağlantı üzerinden bu TCP'nin gönderdiği son veribölütünün sıra numarasına eşit bir değer aldığını varsayalım. Tam o anda konağın çöküp kurtarıldığını ve bağlantının yeni bir varoluşunun kurulduğunu varsayalım. Seçilen ilk sıra numarası S1 = İSN(t), bağlantının eski varoluşunda kullanılan son sıra numarası olur. Eğer kurtarma yeterince çabuk gerçekleşirse, ağdan S1'in komşu sıra numaralarını taşıyan eski yinelenmişler gelebilir ve bağlantının yeni varoluşunun alıcısı tarafından bunlar yeni paketlermiş gibi ele alınabilir.

Burada sorun, konağın kurtarılıncaya kadan ne kadar çökük kaldığının ve önceki bağlantı varoluşundan sistemde hala eski yinelenmişlerin kalıp kalmadığının bilinmeyişidir.

Bu sorunu aşmanın tek yolu, bir çökme sonucunda kurtarmanın ardından gönderilecek veribölütlerine kasıtlı olarak bir AVÖ'lük gecikme uygulamaktır (bu bir "sessizlik süresi" belirtimidir). Belli bir hedef için eski ve yeni paketlerin karışma olasılığı riskini almaya istekli olup beklemekten kaçınmayı tercih eden konaklar, "sessizlik süresi" için beklemeyi seçmeyebilirler. Gerçeklenimciler TCP kullanıcılarına, bağlantıdan bağlantıya bir çökme sonrası beklenip beklenmeyeceğini veya tüm bağlantılar için "sessizlik süresi"nin teklifsizce mi gerçekleneceğini seçme imkanı verebilirler. Besbelli, bir kullanıcı "beklemeyi" seçtiğinde bile, en azından AVÖ saniye sonra "ayağa kalkmış" bir konak için bu gerekli değildir.

Özet olarak: yayımlanan her veribölütü, sıra numarası uzayındaki bir veya daha fazla sıra numarasını işgal eder; bir veribölütü tarafından işgal edilen numaralar AVÖ saniyece "kullanımda" ya da "meşgul"dürler; son yayımlanan veribölütünün sekizlileri tarafından işgal edilmiş bir uzay–zaman blokunun çöküşünün ardından, eğer yeni bağlantı hemen başlatılır ve önceki bağlantı varoluşunun son veribölütünün uzay–zaman yayılımındaki sıra numaralarından biri kullanılırsa alıcı tarafta bir karışıklığa sebep olabilecek bir sıra numarası çakışması olasılığı var demektir.

3.4. Bağlantının Kurulması

Bir bağlantı kurmak için "üçlü uzlaşım" yöntemi kullanılır. Bu yöntem normal olarak bir TCP tarafından başlatılır, diğer TCP tarafından yanıtlanır. Bu yöntem iki TCP'nin de aynı anda yöntemi başlatması durumunda da iş görür. Aynı anda başlatma durumunda, her TCP bir "EŞZ" alındıktan sonra gönderilmesi gereken alındının olmadığı, sadece "EŞZ" taşıyan birer veribölütü alır. Şüphesiz, aynı anda bağlantı başlatma durumunda alıcı tarafa eski bir yinelenmiş "EŞZ" veribölütü gelme olasılığı da vardır. "Baştan–başlat" veribölütleri bu gibi durumlarda kullanılarak sorun aşılabilir.

Aşağıda birkaç tane bağlantı başlatma örneği verilmiştir. Bu örneklerde, alıcı TCP'nin veriyi geçerli olduğundan emin oluncaya kadar teslim etmediği veri taşıyan veribölütleri kullanılarak yapılan bir bağlantı eşzamanlaması

gösterilmemiş olsa da, bu tamamen meşrudur. (Gösterilse de, bağlantı KURULU durumuna geçinceye kadar verinin tamponlanması gerekecekti.) Üçlü uzlaşım yöntemi yanlış bağlantı olasılığını azaltır. Bu sınama için ise bilgi sağlayan iletilerle bellek arasındaki ödünleşimin gerçeklenimidir.

En basit üçlü uzlaşım *Bağlantı Eşzamanlamasında Üçlü Uzlaşımın En Basit Kullanımı* (sayfa: 23)'de gösterilmiştir. Şekildeki gösterim şöyle yorumlanmalıdır: Her satır atıf yapılabilmesi için numaralanmıştır. Sağa ok (-->) ile TCP A'dan TCP B'ye bir TCP veribölütünün gidişi veya bir veribölütünün A'dan B'ye varışı gösterilmiştir. Sola ok (<--) ise tersini belirtir. Nokta üçlemeleri (...) bir veribölütünün hala ağda olduğunu (geciktiğini), bir "XXX" kaybolan veya reddedilen bir veribölütünü gösterir. Açıklamalar parantez içinde gösterilmiştir. TCP durumları bir veribölütünün gidişi veya varışından SONRA ki durumu gösterir (içerikler her satırın ortasında gösterilmiştir). Veribölütü içerikleri kısaltmalar kullanılarak sıra numarası, denetim bayrakları ve alındı alanı ile gösterilmiştir. Pencere, adresler, uzunluklar ve metin gibi alanlar duru bir görünüm yararına dışarda bırakılmıştır.

Şekil 7. Bağlantı Eşzamanlamasında Üçlü Uzlaşımın En Basit Kullanımı

```
      TCP A                                                TCP B

  1.  KAPALI                                               DİNLE

  2.  EŞZ-GÖNDER --> <SIRA=100><DNT=EŞZ>               --> EŞZ-ALINDI

  3.  KURULU     <-- <SIRA=300><ALN=101><DNT=EŞZ,ALN>   <-- EŞZ-ALINDI

  4.  KURULU     --> <SIRA=101><ALN=301><DNT=ALN>       --> KURULU

  5.  KURULU     --> <SIRA=101><ALN=301><DNT=ALN><VERİ> --> KURULU

  Bağlantı eşzamanlamasını sağlamak için üçlü uzlaşımın basit kullanımı
```

Bağlantı Eşzamanlamasında Üçlü Uzlaşımın En Basit Kullanımı (sayfa: 23)'nin 2. satırında TCP A 100 ile başlayan sıra numaraları kullanacağını belirten bir EŞZ veribölütü göndererek uzlaşımı başlatır. 3. satırda, TCB B, TCB A'dan bir EŞZ aldığını belirten bir alındı (ALN) ve bir EŞZ gönderir. TCB B'nin EŞZ'yi 100 sıra numarası ile kabul ettiğini ve alındı alanında sıra numarası olarak 101 belirterek 101 sıra numaralı sekizliyi beklediğini belirttiğine dikkat ediniz.

4. satırda, TCP A, TCP B'nin EŞZ'si için bir ALN içeren verisiz bir veribölütüyle yanıt verir. 5. satırda ise, TCP A bir miktar veri yollar. 5. satırdaki veribölütünün sıra numarasının 4. satırdaki ile aynı oluşuna dikkat ediniz. Bunun sebebi ALN'nin sıra numarası uzayında bir yer işgal etmeyişidir (Etseydi, bir de ALN alındılaşmak için uğraşacaktık).

Aynı anda başlatma *Aynı Anda Bağlantı Eşzamanlaması* (sayfa: 23)'de gösterildiği üzere biraz daha karmaşıktır. Her TCP sırayla KAPALI, EŞZ-GÖNDER, EŞZ-ALINDI, KURULU durumlarına geçer.

Şekil 8. Aynı Anda Bağlantı Eşzamanlaması

```
      TCP A                                            TCP B

  1.  KAPALI                                           KAPALI

  2.  EŞZ-GÖNDER --> <SIRA=100><DNT=EŞZ>           ...

  3.  EŞZ-ALINDI <-- <SIRA=300><DNT=EŞZ>           <-- EŞZ-GÖNDER
```

```
4.              ... <SIRA=100><DNT=EŞZ>              --> EŞZ-ALINDI

5.  EŞZ-ALINDI --> <SIRA=100><ALN=301><DNT=EŞZ,ALN> ...

6.  KURULU     <-- <SIRA=300><ALN=101><DNT=EŞZ,ALN> <-- EŞZ-ALINDI

7.              ... <SIRA=101><ALN=301><DNT=ALN>     --> KURULU
```

Bağlantının aynı anda eşzamanlanması

Üçlü uzlaşımın başlıca sebebi, eski yinelenmiş bağlantı ilklendirmelerinin karışıklığa neden olmasını önlemektir. Bunu sağlayabilmek için özel bir denetim iletisi, baştan–başlat (BŞT) icadedilmiştir. Eğer alıcı TCP eşzamanlama yapılan durumlardan birinde (EŞZ-GÖNDER, EŞZ-ALINDI) ise makul bir BŞT aldığında DİNLE durumuna döner. Eğer alıcı TCP, eşzamanlanmış durumlardan birinde (KURULU, SON-BEKLE-1, SON-BEKLE-2, KAPAT-BEKLE, KAPANIŞ, SON-ALN, ZMN-BEKLE) ise bağlantıyı terkeder ve kullanıcıyı bilgilendirir. Bu son durumu daha ileride "yarı açık" bağlantılar olarak inceleyeceğiz.

Şekil 9. Eski EŞZ Kopyasından Kurtulma

```
     TCP A                                                TCP B

1.   KAPALI                                              DİNLE

2.   EŞZ-GÖNDER   --> <SIRA=100><DNT=EŞZ>              ...

3.   (eski kopya) ... <SIRA=90><DNT=EŞZ>               --> EŞZ-ALINDI

4.   EŞZ-GÖNDER   <-- <SIRA=300><ALN=91><DNT=EŞZ,ALN>  <-- EŞZ-ALINDI

5.   EŞZ-GÖNDER   --> <SIRA=91><DNT=BŞT>               --> DİNLE

6.                ... <SIRA=100><DNT=EŞZ>              --> EŞZ-ALINDI

7.   EŞZ-GÖNDER   <-- <SIRA=400><ALN=101><DNT=EŞZ,ALN> <-- EŞZ-ALINDI

8.   KURULU       --> <SIRA=101><ALN=401><DNT=ALN>     --> KURULU
```

Eski EŞZ Kopyasından Kurtulma

Basit bir eski kopyalardan kurtulma örneği *Eski EŞZ Kopyasından Kurtulma* (sayfa: 24)'da ele alınmıştır. 3. satırda, TCB B'ye eski bir EŞZ kopyası gelmektedir. TCB B bunun eski bir kopya olduğunu bilemez ve normal şekilde yanıtlar (4. satır). TCP A, ALN alanının yanlış olduğunu saptar ve veribölütünü güvenilir kılacak şekilde seçilmiş bir SIRA alanıyla bir BŞT (baştan–başlat) döndürür. TCB B, BŞT aldığında DİNLE durumuna geçer. Son defa olarak özgün EŞZ 6. satırda gönderildiğinde eşzamanlama normal olarak başlar. Eğer 6. satırdaki EŞZ, BŞT'den önce B'ye varırsa BŞT'nin göndericisiyle her iki yönde biraz daha karmaşık bir değiştokuş oluşabilir.

3.4.1. Yarı Açık Bağlantılar ve Diğer Aykırılıklar

Eğer TCP'lerden biri diğer ucun haberi olmaksızın kapanmış veya bağlantıyı terketmişse ya da bellek kaybı ile sonuçlanan bir çökme sonucu bağlantının iki ucu arasındaki eşzamanlama bozulmuşsa bağlantı yarı açıktır denir. Böyle bağlantılarda, uçlardan biri veri göndermeye çalışırsa uzlaşım kendiliğinden baştan başlatılır. Yine de, yarı açık bağlantılar olağandışı durumlardan olup kurtarma işlemi de sessizce yapılır.

Eğer A tarafında bağlantı artık mevcut değilse ve B tarafındaki kullanıcı bir veri göndermeye çalışırsa, işlem, B tarafındaki TCP'nin bir baştan–başlat denetim iletisi almasıyla sonuçlanacaktır. Böyle bir ileti B tarafındaki TCP'nin bir şeyleri yanlış yaptığını belirtir ve bağlantıyı terketmesi beklenir.

A ve B isimli iki kullanıcı süreci iletişim halindeyken A'da bellek kaybına sebep olan bir çökmenin yaşandığını varsayalım. İşletim sisteminin A'nın TCP'sini destekleyişine bağlı olarak bir takım hatadan kurtulma mekanizmaları mevcut olabilir. TCP tekrar ayağa kalktığında, A, görevine baştan ya da kurtarma noktasından başlayacaktır. Sonuç olarak A, muhtemelen bağlantıya ya tekrar bir **AÇ**ma işlemi uygulayacak veya açık olduğunu sandığı bağlantıdan **GÖNDER**me işlemini deneyecektir. İkinci durumda, yerel (A'nın) TCP'den "bağlantı açık değil" şeklinde bir hata iletisi alır. Bir bağlantı kurmak için A'nın TCP'si EŞZ içeren bir veribölütü gönderir. Senaryo *Yarı Açık Bağlantının Keşfi* (sayfa: 25)'da gösterildiği gibi sürüp gider. TCP A'nın çöküşü sonrası, kullanıcısı bağlantıyı yeriden açmaya çalışırken, TCB B'nin kullanıcısı ise bağlantının açık olduğunu düşünmektedir.

Şekil 10. Yarı Açık Bağlantının Keşfi

```
      TCP A                                                TCP B

  1.  (ÇÖKÜŞ)                                    (gönderi 300, alım 100)

  2.  KAPALI                                           KURULU

  3.  EŞZ-GÖNDER --> <SIRA=400><DNT=EŞZ>          --> (??)

  4.  (!!)       <-- <SIRA=300><ALN=100><DNT=ALN> <-- KURULU

  5.  EŞZ-GÖNDER --> <SIRA=100><DNT=BŞT>          --> (Terket!!)

  6.  EŞZ-GÖNDER                                       KAPALI

  7.  EŞZ-GÖNDER --> <SIRA=400><DNT=EŞZ>          -->

                Yarı Açık Bağlantının Keşfi
```

3. satırda EŞZ vardığında, TCP B eşzamanlanmış durumdaydı ve gelen veribölütü pencere dışındaydı. Verilecek yanıt, beklediği sıra numarasını belirten bir alındı (ALN 100) göndermek olacaktı; öyle yaptı. TCP A'nın gördüğü ise gönderdiği hiçbirşeyin alınmamış olduğuydu ve üstelik eşzamanlama da yoktu; bir baştan–başla (BŞT) gönderdi, çünkü bağlantının yarı açık olduğunu saptamıştı. TCP B'nin, 5. satırda bağlantıyı terkettiğini, TCP A'nın ise bağlantıyı kurmaya çalıştığını görürüz. Artık sorun *Bağlantı Eşzamanlamasında Üçlü Uzlaşımın En Basit Kullanımı* (sayfa: 23)'deki üçlü uzlaşıma indirgenmiştir.

TCP A çökerken TCP B'nin eşzamanlı olduğunu düşündüğü bağlantı üzerinden veri göndermeyi denediği durumla ilgili ilginç bir durum daha vardır. Bu, *Yarı Açık Bağlantının Açık Taraftan Keşfedilmesi* (sayfa: 25)'de gösterilmiştir. Bu durumda, TCP A'ya TCP B'den gelen veri (2. satır) kabul edilemez, çünkü artık bağlantı mevcut değildir; dolayısıyla TCP A bir BŞT gönderir. BŞT kabul edilebilir olduğundan TCP onu işleme sokar ve bağlantıyı terkeder.

Şekil 11. Yarı Açık Bağlantının Açık Taraftan Keşfedilmesi

```
       TCP A                                                TCP B

  1.  (ÇÖKÜŞ)                                    (gönderi 300, alım 100)

  2.  (??)     <-- <SIRA=300><ALN=100><VERİ=10><DNT=ALN> <-- KURULU
```

```
3.          --> <SIRA=100><DNT=BŞT>                    --> (Terket!!)

          Yarı Açık Bağlantının Açık Taraftan Keşfedilmesi
```

Eski EŞZ kopyasının iki Edilgen Soketi İlklendirmesi (sayfa: 26)'de EŞZ bekleyen edilgen bağlantılı A ve B TCP'lerini görüyoruz. TCP B'ye 2. satırda eski bir yinelenmiş gelerek TCP B'yi telaşlandırıyor. Bir EŞZ–ALN göndererek (3. satırda) TCP A'nın bir BŞT üretmesine sebep oluyor (3. satırdaki ALN kabul edilebilir değildir). TCP B baştan başlamayı kabul ediyor ve tekrar edilgen DİNLEme durumuna geri dönüyor.

Şekil 12. Eski EŞZ kopyasının iki Edilgen Soketi İlklendirmesi

```
     TCP A                                              TCP B

1.   DİNLE                                              DİNLE

2.         ... <SIRA=Z><DNT=EŞZ>              -->   EŞZ-ALINDI

3.   (??)  <-- <SIRA=X><ALN=Z+1><DNT=EŞZ,ALN> <--   EŞZ-ALINDI

4.         --> <SIRA=Z+1><DNT=BŞT>            -->   (DİNLEme durumuna geç!)

5.   DİNLE                                              DİNLE

     Eski EŞZ Kopyasının iki Edilgen Soketi İlklendirmesi
```

Başka durumlar da olasıdır, hesaba katılanlardan BŞT üretimi ve işlenmesi ile ilgili olanlar aşağıdadır.

3.4.2. Baştan Başlatmanın İstenmesi

Genel bir kural olarak, baştan başlatma (BŞT) gelen bir veribölütünün mevcut bağlantı için tasarlanmamış olması halinde gönderilmelidir. Bu durum açıkça belli değilse bir baştan başlatma gönderilmemelidir.

Üç grup durum vardır:

1. Bir bağlantı mevcut değilse (KAPALI), gelen bir veribölütüne bir BŞT içermiyorsa yanıt olarak bir BŞT gönderilir. Özellikle iki tarafın da başlatmadığı mevcut olmayan bir bağlantıya gelen EŞZler bu anlamda reddedilirler.

   Gelen veribölütü bir ALN alanına sahipse, baştan başlatma için sıra numarası veribölütünün ALN alanından alınır, aksi takdirde başlan başlatma sıra numarası sıfırdır ve ALN alanının değeri gelen veribölütünün veribölütü uzunluğu ile sıra numarasının toplamı olur. Bağlantı KAPALI durumda kalır.

2. Bağlantı eşzamanlama yapılan durumlardan (DİNLE, EŞZ-GÖNDER, EŞZ-ALINDI) birinde değilse ve gelen veribölütü henüz gönderilmemiş birşeylerin alındığı bilgisini içeriyorsa (veribölütü kabul edilebilir olmayan bir ALN taşıyordur) veya gelen veribölütü bir güvenlik seviyesine ya da seviye ve bağlantı için istenen bölüm ile uyumsuz bir bölüme sahipse bir baştan–başlat gönderilir.

   Bizim EŞZ alındılanmamışsa ve gelen veribölütünün öncelik seviyesi istenen öncelik seviyesinden daha yüksekse ya yerel öncelik seviyesi yükseltilir (sistem veya kullanıcı izin vermişse) ya da BŞT gönderilir; veya gelen veribölütünün öncelik seviyesi istenen öncelik seviyesinden daha düşükse öncelikler eşleşmiş gibi devam edilir (uzak TCP öncelik seviyesini bizimki ile eşleşmek üzere yükseltemezse bunu sonraki veribölütünü gönderirken saptamış olur ve bağlantı sonlandırılır). Eğer bizim EŞZ alındılanmışsa (tabii ki, gelen veribölütünde) ve bir baştan–başlat gönderilmeli değilse gelen veribölütünün öncelik seviyesi ile yerel öncelik seviyesi tam olarak eşleşmelidir.

Eğer gelen veribölütünde bir ALN alanı varsa BŞT için bu veribölütünün ALN alanındaki sıra numarası kullanılır, aksi takdirde, BŞT için sıra numarası sıfır olurken, ALN alanına sıra numarası ile gelen veribölütünün veribölütü uzunluğunun toplamı atanır. Bağlantı mevcut durumunu korur.

3. Eğer bağlantı eşzamanlanmış durumdaysa (KURULU, SON-BEKLE-1, SON-BEKLE-2, KAPAT-BEKLE, KAPANIŞ, SON-ALN, ZMN-BEKLE) kabul edilebilir olmayan bir veribölütü (pencere dışı bir sıra numarası veya kabul edilebilir olmayan bir alındı numarası), sadece o anki gönderi sıra numarasını ve alınması umulan sonraki sıra numarasını belirten bir alındı içeren verisiz bir veribölütüne sebep olmalı ve bağlantı aynı durumda kalmalıdır.

   Eğer gelen veribölütü bağlantı için istenen seviye, bölüm veya önceliğe uymayan bir güvenlik seviyesi, bölümü veya önceliğine sahipse bir BŞT gönderilir ve bağlantı KAPALI duruma döner. BŞT sıra numarası gelen veribölütünün ALN alanından alınır.

3.4.3. Baştan Başlama İşlemi

EŞZ-GÖNDER hariç tüm durumlarda, tüm BŞT veribölütleri SIRA alanlarına bakılarak doğrulanır. Bir baştan başlatmanın geçerli olabilmesi için onun sıra numarası pencere içinde kalmalıdır. EŞZ-GÖNDER durumunda ise (bir ilk EŞZ'ye yanıt olarak bir BŞT alınması) baştan başlatmanın kabul edilebilmesi için ALN alanı EŞZ alındısını içermelidir.

BŞT'nin alıcısı ilk değerlendirmeyi yapar ve durumunu değiştirir. Eğer alıcı DİNLE durumundaysa onu yoksayar. Eğer alıcı EŞZ-ALINDI durumundaysa ve önceki durumu DİNLE durumu ise alıcı DİNLE durumuna geri döner, aksi takdirde, alıcı bağlantıyı terkeder ve KAPALI durumuna geçer. Eğer alıcı bunlar dışında bir durumdaysa bağlantıyı terkeder ve kullanıcıyı bilgilendirip KAPALI duruma geçer.

3.5. Bağlantının Kapatılması

KAPAT, "Göndereceğim veri kalmadı" anlamına gelen bir işlemdir. Bir iki yönlü çalışan bağlantının kapatılması, alıcı tarafın bunu nasıl ele alacağı bilinemediğinden, anlamı belirsiz bir yorumlamanın konusudur, şüphesiz. Biz **KAPAT**'ı tek yönlü bir bağlantıdaymışız gibi ele alacağız. Diğer ucun KAPALI duruma geçtiğini öğrenene kadar **KAPAT**an kullanıcı **AL**maya devam edebilir. Bu durumda, bir program bir **KAPAT** öncesinde bazı **GÖNDER**ileri başlatmış olabilirdi ve öbür ucun KAPALI olmasından dolayı bir **AL**ımın başarısız olduğu sinyallenene kadar **AL**maya devam ederdi. Öbür ucun kapanmasından dolayı ortada hiçbir **AL**ım kalmasa bile, biz TCP'nin kullanıcıyı haberdar edeceğini ve böylece kullanıcının kendi tarafını sonlandırabileceğini varsayıyoruz. Bir TCP, bağlantı KAPALI duruma geçmeden önce tüm **GÖNDER** tamponlarını güvenle teslim edecektir. Böylece dönecek hiçbir veri kalmadığını uman kullanıcının duymayı beklediği tek şey tüm verisinin hedef TCP tarafından tamamen alınmasıyla öbür ucun KAPALI duruma geçtiğidir. TCP artık veri kalmadı diyene kadar kullanıcılar gönderime kapadıkları bağlantılarını okumaya açık tutmalıdırlar.

Başlıca üç durum sözkonusudur:

1. Kullanıcı TCP'ye bağlantıyı **KAPAT** demeye başlar.
2. Uzak TCP bir SON denetim sinyali göndermeye başlar.
3. Her iki kullanıcı aynı anda **KAPAT**ır.

1. durum: **Yerel kullanıcı kapanmayı başlatır**

   Bu durumda, bir SON veribölütü oluşturulur ve giden veribölütü kuyruğuna yerleştirilir. TCP tarafından kullanıcıdan artık bir **GÖNDER** kabul edilmez ve TCP SON-BEKLE-1 durumuna geçer. Bu durumda **AL**ımlara izin verilir. SON içeren ve SON ile öncelenmiş tüm veribölütleri alındılanana kadar yeniden aktarılırlar. Diğer TCP, SON'un her ikisini de alındıladığında ve kendi SON'unu gönderdiğinde, ilk TCP bu SON'u ALN'leyebilir. Bir SON alan TCP'nin bunu ALN'leyeceğini fakat ek olarak kullanıcısının bağlantısı KAPALI olana kadar kendi SON'unu göndermeyeceğine dikkat ediniz.

2. durum: **TCP ağdan bir SON alır**

Eğer ağdan talep edilmemiş bir SON gelirse, alıcı TCP onu ALN'leyebilir ve kullanıcıya bağlantının kapanacağını söyler. TCP'nin kalan veriyi diğer TCP'ye gönderdikten sonra bir SON göndermesi için kullanıcı bunu bir **KAPAT** ile yanıtlar. TCP bundan sonra kendi SON'unun alındılanmasını bekler ve alındıyı aldıktan sonra bağlantıyı siler. Bir ALN ulaşmazsa, bir kullanıcı zamanaşımı sonrasında bağlantı terkedilir ve kullanıcıya haber verilir.

3. durum: **Her iki taraf aynı anda kapanır**

Bir bağlantının her iki ucunun kullanıcılarından aynı anda **KAPAT** gelmesi SON veribölütlerinin değiştokuşuna sebep olur. SONlarla öncelenmiş tüm veribölütleri işleme alınıp alındılandığında her TCP kendi aldığı SON'u ALN'leyebilir. Her ikisi de bu ALN'leri aldıktan sonra bağlantıyı silerler.

Şekil 13. Normal Kapanma

```
      TCP A                                                TCP B

  1.  KURULU                                               KURULU

  2.  (Kapat)
      SON-BEKLE-1 --> <SIRA=100><ALN=300><DNT=SON,ALN> --> KAPAT-BEKLE

  3.  SON-BEKLE-2 <-- <SIRA=300><ALN=101><DNT=ALN>     <-- KAPAT-BEKLE

  4.                                                       (Kapat)
      ZMN-BEKLE   <-- <SIRA=300><ALN=101><DNT=SON,ALN> <-- SON-ALN

  5.  ZMN-BEKLE   --> <SIRA=101><ALN=301><DNT=ALN>     --> KAPALI

  6.  (2 AVÖ)
      KAPALI

                          Normal Kapanma
```

Şekil 14. Aynı Anda Kapanma

```
      TCP A                                                TCP B

  1.  KURULU                                               KURULU

  2.  (Kapat)                                              (Kapat)
      SON-BEKLE-1 --> <SIRA=100><ALN=300><DNT=SON,ALN> ... SON-BEKLE-1
                  <-- <SIRA=300><ALN=100><DNT=SON,ALN> <--
                  ... <SIRA=100><ALN=300><DNT=SON,ALN> -->

  3.  KAPANIŞ     --> <SIRA=101><ALN=301><DNT=ALN>     ... KAPANIŞ
                  <-- <SIRA=301><ALN=101><DNT=ALN>     <--
                  ... <SIRA=101><ALN=301><DNT=ALN>     -->

  4.  ZMN-BEKLE                                            ZMN-BEKLE
      (2 AVÖ)                                              (2 AVÖ)
      KAPALI                                               KAPALI
```

Aynı Anda Kapanma

3.6. Öncelik ve Güvenlik

Burada amaç sadece, tamamıyla aynı güvenlik ve bölme değerli, iki port tarafından da talep edilen öncelik düzeyinden daha yüksek düzeydeki portlar arasında bağlantının olmasına izin verilmesini sağlamaktır

TCP tarafından kullanılan öncelik ve güvenlik parametrelerinin hepsi Genel Ağ Protokolü (IP) [2] içinde tanımlanmıştır. Bu TCP tanımlaması boyunca kullanacağımız "güvenlik/bölüm" terimi ile kastedilen, güvenlik, bölüm, kullanıcı grupları ve sınırlamayı dahilinde bulunduran IP içinde kullanılan güvenlik parametreleridir.

Bir bağlantı girişiminin güvenlik/bölüm değerinin uyuşmaması veya daha düşük öncelikte olması durumunda, bağlantının bir baştan başlat gönderilerek reddedilmesi gerekir. Bir bağlantının çok düşük öncelikli olması nedeniyle reddedilmesi durumu, sadece EŞZ alındısı alındıktan sonra gerçekleşen bir durumdur.

Burada dikkat etmemiz gereken, sadece öntanımlı öncelik değerinde çalışan TCP modülerinin gelen veribölütlerinin öncelik değerine bakmasının ve imkan dahilinde ise bağlantı üzerinde kullandıkları öncelik seviyesinin yükseltilmesinin zorunlu olmasıdır.

Güvenlik parametreleri güvenli olmayan (değerlerin sınıflandırılmamış veri belirtmesi) bir ortamda da kullanılmış olabilirler, bu yüzden güvenli bir ortamda olmayan konaklar güvenlik parametrelerini almaya hazır olmalı, ama tabii ki bu parametreleri göndermek zorunda değildirler.

3.7. Veri İletişimi

Bağlantı bir kere kurulduktan sonra veri iletişimi veribölütlerini değiştokuş ederek gerçekleşir. Hatalar (sağlama özeti sınamalarının başarısız olması), ağda sıkışıklık gibi nedenlerden dolayı veribölütleri kaybolabildiğinden veribölütlerinin her birinin teslimatını garantiye almak için TCP yeniden aktarım (bir zamanaşımı sonrası) yapar. Bu işlemden dolayı veya ağın yapısı gereği yinelenmiş veribölütleri gelebilir. Sıra numaraları ile ilgili bölümde açıklandığı gibi TCP veribölütlerindeki sıra ve alındı numaraları üzerinde kabul edilebilirliklerini doğrulamak için bazı sınamalar yapar.

Verinin göndericisi kullanacağı sonraki sıra numarasının değerini GÖN.SNR değişkeninde saklar. Verinin alıcısı ise almayı umduğu sonraki sıra numarasını ALM.SNR değişkeninde saklar. Verinin göndericisi alındılanmamış en eski sıra numarasını GÖN.OLM değişkeninde tutar. Eğer gönderilmiş tüm veri alındılandıktan sonra veri akışı bir süre için boşta kalırsa üç değişkenin eşitlendiği görülür.

Gönderici bir veribölütünü oluşturup gönderdikten sonra GÖN.SNR'yi arttırır. Alıcı ise veribölütünü kabul ettikten sonra ALM.SNR'yi arttırır ve bir alındı yollar. Bu alındı veri göndericisine ulaştığında GÖN.OLM'yi arttırır. Bu değişkenlerin değerleri arasındaki fark iletişimdeki gecikmenin bir ölçüsüdür. Değişkenlerin arttırım miktarı veribölütündeki verinin uzunluğu kadardır. KURULU duruma geçildikten sonra tüm veribölütlerinin alındı bilgilerine dayanarak taşınması gerektiğine dikkat ediniz.

KAPAT kullanıcı çağrısı, bir gelen veribölütündeki SON denetim bayrağının yaptığı gibi bir gitsin işlemi uygular.

3.7.1. Yeniden Aktarım Zamanaşımı

Ağlar arası sistemi oluşturan ağın değişkenliği ve TCP bağlantılarının geniş çapta kullanımı sebebiyle yeniden aktarım zamanaşımının özdevimli saptanması gerekir. Burada göstermelik olarak bir yeniden aktarım zamanaşımını saptama işlemi örneklenmiştir.

Örnek 1. Bir Yeniden Aktarım Zamanaşımını Saptama İşlemi Örneği

Beli bir sıra numarası ile bir verinin gönderilmesi ile bu sıra numarasını kapsayan bir alındının alınması (gönderilen veribölütleri ile alınanların eşleşmesi zorunlu değildir) arasında geçen süre ölçülür. Ölçülen bu süreye Gidip Gelme Zamanı (GGZ) denir. Bundan Yaklaşık Gidip Gelme Zamanı (YGGZ) hesaplanır:

```
YGGZ = ( ALFA * YGGZ ) + ((1-ALFA)) * GGZ)
```

buna bağlı olarak da yeniden aktarım zamanaşımı (YAZ) hesaplanır:

```
YAZ = asg[ÜSINIR,azm[ASINIR,(BETA*YGGZ)]]
```

Burada ÜSINIR zamanaşımının üst sınırını (örn, 1 dakika), ASINIR ise zamanaşımının alt sınırını (örn, 1 saniye) ifade etmektedir. ALFA bir yuvarlatma katsayısı (örn, 0,8 ... 0,9), BETA ise gecikme değişim katsayısıdır (örn, 1,3 ... 2,0).

3.7.2. Acil Bilgi İletişimi

TCP acil durum mekanizmasının amacı gönderen kullanıcının alan kullanıcıyı acil veriyi kabul etmesi için teşvik etmek ve alan kullanıcının bütün acıl veriyi alması durumunda alan TCP'nin bunu belirtmesine izin vermeyi mümkün kılmaktır.

Bu mekanizma veri akımındaki bir noktanın acil bilgi sonu olarak tasarlanmasına izin verir. Alıcı TCP'de alım sıra numarasının (ALM.SNR) bu noktanın her önüne geçişinde, TCP kullanıcıya "acil kip"e geçmesini, alım sıra numarası aciliyet göstericisini yakaladığında ise "normal kip"e geçmesini söylemelidir. Eğer aciliyet göstericisi kullanıcı "acil kip"te iken güncellenirse, güncelleme kullanıcıya görünür olmayacaktır.

Bu yöntem, aktarılan tüm veribölütlerinde taşınmak üzere bir aciliyet alanı tahsis eder. ACL denetim bayrağı aciliyet alanının anlamlı olduğunu belirtir ve aciliyet göstericisini belirtmek üzere veribölütü sıra numarasına eklenmesi gerekir. Bu bayrağın yokluğu ortada acil bir verinin bulunmadığını gösterir.

Bir aciliyet göstericisi göndermek için kullanıcı ek olarak en azından bir veri sekizlisi göndermek zorundadır. Eğer gönderen kullanıcı ayrıca bir gitsin belirtirse hedef sürece acil bilginin teslim zamanı ileri alınır.

3.7.3. Pencere Yönetimi

Her veribölütünde gönderilen pencere, pencere göndericisinin (veri alıcısının) o an kabul etmeye hazır olduğu sıra numaraları aralığını gösterir. Bu bağlantıdaki veri tamponunun kullanılabilir alanı ile ilgili bir önkabul vardır.

Geniş bir pencere belirtilmesi aktarımı teşvik eder. Kabul edilebilecekten fazla veri gelmesi durumunda ise fazlası iptal edilir. Bu, haddinden fazla yeniden gönderime, ağa ve TCP'lere gereksiz yük eklenmesine sebep olur. Küçük bir pencere belirtilmesi, veri aktarımını, veribölütü aktarımları arasındaki gidip gelme gecikmesini başlatan noktada sınırlayabilir.

Sağlanan mekanizmalar bir TCP'ye büyük bir pencerenin ardından daha fazla veri kabul edilmesini zorlamaksızın daha küçük bir pencere ilan etmesini mümkün kılar. Buna "percere kırpma" denir ve kesinlikle mani olunur. Sağlamlık ilkesi gereğince, TCP'ler kendi pencerelerini kırpamazlar, fakat diğer TCP'lerde kısmen böyle bir davranışa karşı hazırlıklı olunmalıdır.

Gönderen TCP kullanıcıdan geleni kabule hazır olmalı ve gönderi penceresi sıfır bile olsa en azından yeni verinin bir sekizlisini göndermelidir. Gönderen TCP pencere sıfır olsa bile alıcı TCP'ye düzenli olarak veriyi yeniden aktarmalıdır. Percerenin sıfır olduğu durum için yeniden aktarım aralığı olarak 2 dakika önerilmektedir. Bu yeniden aktarımda aslolan sıfır pencereli TCP'lerle pencerenin yeniden her açılışının diğerine güvenilir şekilde raporlanacağını garanti etmektir.

Alan TCP'nin bir sıfır penceresi varken bir veribölütü geldiğinde, mevcut pencereyi (sıfır) ve sonraki beklediği sıra numarasını gösteren bir alındıyı yine de göndermelidir.

Gönderen TCP aktaracağı veriyi o anki pencereye sığacak şekilde veribölütleri olarak paketler ve yeniden aktarım kuyruğunda bu veribölütlerini yeniden paketleyebilir. Böyle yeniden paketleme gerekli değildir ama yararlı olabilir.

Veri akışının tek yönlü olduğu bir bağlantıda, pencere bilgisi hepsi aynı sıra numarasına sahip alındı veribölütlerinde taşınır, bu durumda sırasız bile gelseler onları yeniden sıralamak için hiçbir yol olmayacaktır. Bu ciddi bir sorun değildir, fakat pencere bilgisinin ara sıra geçici olarak veri alıcıdaki eski raporları temel alması mümkün olacaktır. Bu sorundan kaçınmak için en yüksek alındı numarasını taşıyan veribölütlerindeki (evvelce alınmış en yüksek alındı numarasına eşit veya daha büyük alındı numaralı veribölütlerindeki) pencere bilgisine göre hareket etmek en iyisidir.

Pencere yönetim yöntemi, iletişim başarımında önemli bir etkiye sahiptir. Aşağıdaki açıklamalar, gerçeklenimcilere önerilerdir.

3.7.3.1. Pencere Yönetim Önerileri

Az sayıda büyük veribölütü kullanarak daha iyi başarımın elde edildiği durumda çok küçük bir pencere ayrılması verinin çok sayıda küçük veribölütünde aktarılmasına sebep olur.

Küçük pencerelerden kaçınmak için bir öneri, alıcı için bir pencerenin güncellenmesini, bağlantı için olası en büyük tahsisatın en azından yüzde X'i kadar ek bir tahsis gerçekleşene kadar ertelemektir (burada X, 20 ile 40 arası olabilir).

Diğer bir öneri ise, göndericinin veriyi göndermeden önce pencerenin yeterince büyümesini bekleyerek küçük veribölütleri göndermekten kaçınmasıdır. Kullanıcıdan bir gitsin işlemi gelirse veribölütü küçük bile olsa veri gönderilmelidir.

Alındıların geciktirilmemesi gerektiğine yoksa gereksiz yeniden aktarıma yolaçılacağına dikkat ediniz. Bir strateji küçük bir veribölütü geldiğinde (pencere bilgisini güncellemeksizin) bir alındı gönderdikten sonra pencere büyük olduğunda yeni pencere bilgisi ile başka bir alındı göndermek olurdu.

Sıfır penceresi algılanmak üzere gönderilmiş veribölütü ek olarak aktarılan verinin giderek daha küçük veribölütlerine parçalanmasını da başlatabilir. Sıfır penceresi algılanmak üzere gönderilmiş tek bir veri sekizlisi içeren bir veribölütü kabul edilirse mevcut pencerenin tek sekizlisini tüketir. Gönderen TCP basitçe sıfırdan farklı pencere gönderdikçe aktarılan veri büyüklü küçüklü veribölütlerine bölünecektir. Zamanla, büyük veribölütlerinin biri biraz küçük diğeri biraz büyük bir çifte bölünmesi pencere tahsis eden alıcıda arasıra beklemelerle sonuçlanır. Ve bir süre sonra veri aktarımı bilhassa küçük veribölütlerinde olmaya başlar.

Burada öneri; en basitini yapma eğilimindeki gerçeklenimlerde pencere yönetim mekanizmaları çok sayıda küçük pencereyle sonuçlanmak eğiliminde olduğundan, TCP gerçeklenimlerinin etkin olarak küçük pencere tahsislerini daha büyük pencereler halinde birleştirmeye çalışmaları gerektiğidir.

3.8. Arayüzler

Şüphesiz işin iki arayüzü vardır: kullanıcı/TCP arayüzü ve TCP/düşük seviye arayüzü. Kullanıcı/TCP arayüzünün tarafsız olarak özenle hazırlanmış bir modeline sahibiz, fakat düşük seviyeli protokolle olan arayüzü, o protokolün belirtimi tarafından belirtileceğinden burada belirtmeden bırakacağız. Daha düşük seviyeli protokolün IP olduğu durumda, TCP'lerin kullanabileceği bazı parametre değerlerine dikkat çekeceğiz.

3.8.1. Kullanıcı/TCP Arayüzü

TCP için kullanıcı komutlarının aşağıdaki işlevsel açıklamaları, her işletim sistemi farklı oluşumlara sahip olacağından, olsa olsa kurgusaldır. Bu bakımdan, okuyucuyu farklı TCP gerçeklenimlerinin farklı kullanıcı arayüzleri olacağı konusunda uyarmamız gerekir. Yine de, tüm TCP gerçeklenimlerinin aynı protokol hiyerarşisini

destekleyebilmesini garantiye almak için tüm TCP'lerin küçük de olsa aynı ortak hizmet kümesini barındırması gerekir. Bu bölümde tüm TCP gerçeklenimlerinin bulundurması gerektiği işlevsel arayüzler belirtilmiştir.

3.8.1.1. TCP Kullanıcı Komutları

Aşağıdaki bölümler işlevsel olarak bir Kullanıcı/TCP arayüzünü betimler. Kullanılan sözdizimi daha çok yüksek seviyeli dillerdeki işlev veya yöntem çağrılarınınkine benzemekle birlikte, bu kullanım tuzak türü hizmet çağrılarının (SVC'ler, UUO'lar, EMT'ler gibi) hariç tutulduğu anlamına gelmez.

Aşağıda açıklanan kullanıcı komutları TCP'nin süreçlerarası iletişimi desteklemek için uygulaması gereken temel işlevleri belirtir. Gerçeklenimlerin her biri bunların kendilerine özgü biçimlerini tanımlamalı ve temel işlevleri birarada veya bir alt küme olarak sağlamalıdırlar. Kısmen, bazı gerçeklenimler belli bir bağlantıda kullanıcı tarafından yapılan ilk **GÖNDER** veya **AL** çağrısıyla bir bağlantının kendiliğinden **AÇ**ılmasını isteyebilirler.

Süreçlerarası iletişim oluşumlarını sağlamada, TCP sadece komutları kabul etmekle kalmamalı ek olarak hizmet sunduğu sürece bilgi de döndürmelidir. Sonuncusu şunlardan oluşur:

a. bir bağlantı hakkında genel bilgi (kesmeler, uzaktan kapanma, belirsiz yabancı soketlerin bağlanması gibi).

b. kullanıcı komutlarına özgü başarı veya çeşitli türde başarısızlık belirten dönüşler..

AÇ

```
AÇ (yerel–port, yabancı–soket, etkin/edilgen
      [, zamanaşımı] [, öncelik] [, güvenlik/bölüm] [, seçenekler])
  -> yerel–bağlantı–ismi
```

Burada yerel TCP'nin hizmet verdiği sürecin farkında olduğunu ve belirtilen bağlantıyı kulanacak olan sürecin yetkili olup olmadığını sınayabileceğini farzediyoruz. TCP'nin gerçeklenime bağlı olarak yerel ağ ve kaynak adresi için TCP tanıtları ya TCP ya da düşük seviyeli protokol (IP gibi) tarafından sağlanır. Bu değerlendirmeler güvenlik kaygısının sonucudur, o kadar ki, hiçbir TCP'nin bir diğeri gibi davranması mümkün değildir. Benzer şekilde, hiçbir süreç TCP'nin hoşgörüsü olmaksızın başka bir süreç gibi davranamaz.

etkin/edilgen seçeneği edilgen olarak belirtilirse bu, gelen bir bağlantı için bir DİNLE çağrısıdır. Bir edilgen açık ya belli bir bağlantıyı bekleyen tamamen belirli bir yabancı soketin ya da herhangi bir bağlantıyı bekleyen belirsiz bir yabancı soketin varlığı ile tanınır. Tamamen belirli bir edilgen çağrı ardından bir **GÖNDER** çalıştırılarak etkin yapılabilir.

Bir aktarım denetim bloğu (ADB) oluşturulur ve **AÇ** komutunun parametrelerindeki veri ile kısmen doldurulur.

Etkin bir **AÇ** komutunda TCP bağlantıyı eşzamanlama işlemini hemen başlatacaktır.

Bir zamanaşımı varsa, çağrıcının TCP'ye teslim edilen tüm veri için bir zamanaşımı belirlemesini sağlar. Eğer veri zamanaşımı süresinde hedefe başarıyla teslim edilmezse TCP bağlantıyı terkedecektir. Şimdiki genel öntanımlı zamanaşımı değeri beş dakikadır.

TCP veya bir işletim sistemi bileşeni, kullanıcıların belirli bir öncelik veya güvenlik/bölüm ile bir bağlantı açma yeterliliğini doğrulayacaktır. **AÇ** çağrısında öncelik veya güvenlik/bölüm belirtiminin yokluğu öntanımlı değerlerin kullanılmasının gerektiğini gösterir.

TCP'nin gelen istekleri kabul etmesi için, güvenlik/bölme bilgilerinin tamamiyle aynı ve önceliğin **AÇ** çağrısında istenen öncelikten büyük veya eşit olması gerekir.

AÇ çağrısında istenen ve gelen bir istekten alınan en büyük değerlerden yüksek olanı bağlantının öncelik değeridir ve bağlantının yaşamı boyunca sabittir. Gerçeklenimciler bu öncelik müzakeresinin denetimini kullanıcıya vermeyi isteyebilirler. Örneğin, kullanıcıya önceliğin tamamen uyuşması gerektiğini belirtebilmesi veya önceliğin terfisine kullanıcı tarafından onay verilmesi mümkün kılınabilir.

TCP tarafından kullanıcıya bir yerel bağlantı ismi döndürülür. Yerel bağlantı ismi `<yerel soket, yabancı soket>` çifti tarafından tanımlanın bağlantıya bir kısayol olarak kullanılabilir.

GÖNDER

GÖNDER (*yerel–bağlantı–ismi*, *tampon–adresi*, *bayt–sayısı*, *gitsin–bayrağı*, *aciliyet–bayrağı* [,*zamanaşımı*])

Bu çağrı belirtilen kullanıcı tamponundaki verinin gönderilmesini sağlar. Eğer bağlantı daha önceden açılmamışsa, **GÖNDER** çağrısı bir hata olduğunu kabul eder. Bazı gerçeklenimler kullanıcının yaptığı ilk çağrının **GÖNDER** çağrısı olmasına izin verir; bu durumda gerçeklenim **AÇ** çağrısının yapılmasını kendisi sağlar. Eğer çağıran süreç bu bağlantıyı kullanmaya yetkili değilse bir hata döner.

Eğer *gitsin–bayrağı* belirtilmişse verinin alıcıya hemen aktarılması ve tampondan oluşturulan son TCP veribölütünde GİT bitinin etkin olması gerekir. Eğer *gitsin–bayrağı* yoksa aktarımın verimliliği adına veri ardışık **GÖNDER** çağrılarındaki veriler birleştirilerek aktarılabilir.

Eğer *aciliyet–bayrağı* belirtilmişse hedef TCP'ye gönderilen veribölütlerinde aciliyet göstericisi bulunur. Alan TCP, eğer acil göstericisinin önceliği veri alıcı süreç tarafından henüz tüketilmediyse alıcı sürece aciliyet durumunu bildirecektir. Aciliyetin amacı, alıcıyı acil veriyi işleme almaya teşvik etmek ve acil olduğu bilinen tüm veri alındığında bunu alıcıya belirtmektir. Gönderen tarafın TCP'sinin sinyallediği aciliyet sayısının, alan kullanıcının acil verinin varlığını öğrendiği uyarı sayısına eşit olması gerekli değildir.

AÇ çağrısında yabancı soket belirtilmemişse fakat bağlantı kurulmuşsa (örn, yerel sokete yabancı bir veribölütünün gelişinden dolayı DİNLEnen bağlantının etkin duruma geçmesi), tasarlanan tampon örtük yabancı sokete gönderilir. Belirsiz yabancı soketli **AÇ** çağrısı yapan kullanıcılar yabancı soketin adresini bilmeksizin **GÖNDER** çağrısı yapabilirler.

Bununla birlikte, yabancı soket belirli duruma gelmeden bir **GÖNDER** çağrısı yapılırsa bir hata dönecektir. Kullanıcılar bağlantının durumunu öğrenmek için **DURUM** çağrısını kullanabilirler. Bazı gerçeklenimlerde TCP belirsiz bir soket bağlandığında kullanıcıyı uyarabilmektedir.

Bir zamanaşımı belirtilmeşse bağlantıya ait zamanaşımı yerine bu değer kullanılır.

En basit gerçeklenimde, **GÖNDER** çağrısı aktarımı tamamlamadan veya zamanaşımına uğramadan denetimi gönderen sürece bırakmayacaktır. Bununla birlikte bu basit yöntemi iki tarafın aynı anda kullanması kısırdöngüye yol açabileceğinden (örn, her iki taraf da bir **AL** çağrısı yapmaksızın **GÖNDER** çağrısı deneyebilirler ve ikisi de birbirlerini bekler) bu önerilmez. Biraz daha karmaşık gerçeklenimlerde çağrı beklemeksizin denetimi sürece bırakarak sürecin ağ G/Ç'leri ile eşzamanlı çalışmasını ve dolayısıyla çok sayıda **GÖNDER** çağrısının yapılabilmesini mümkün kılar. Çok sayıda **GÖNDER** çağrısı ilk gelen ilk gider (FIFO) ilkesiyle işlenir, yani TCP bunları hemen işleme alamaz, kuyruğa ekler.

Bir **GÖNDER** çağrısının sonradan bazı SİNYAL çeşitlerine veya sözde kesmelere yol açması nedeniyle örtük olarak eşzamansız bir kullanıcı arayüzü kabulü yaparız. Diğer bir seçenek hemen bir yanıtın dönmesidir. Örneğin, gönderilen veribölütü uzak TCP tarafından alındılanmamışken bile **GÖNDER** çağrıları beklemeksizin yerel alındılar döndürebilirler. İyi niyetli olarak eninde sonunda bir başarının gerçekleşeceğini varsayabiliriz. Eğer biz yanlışsak, bağlantı her halükarda zamanaşımından dolayı kapanacaktır. Bu çeşit (eşzamanlı) gerçeklenimlerde, hala bazı eşzamansız sinyaller olacaktır fakat bunlar bağlantının kendisi tarafından bertaraf edilecek, veribölütlerine veya tamponlara etkisi olmayacaktır.

Sürecin farklı **GÖNDER** çağrılarından dönen başarı ve hata belirteçlerini ayırması için çağrılarla ilgili tampon adreslerine bakılır. TCP tarafından kullanıcıya verilen sinyaller, çağıran sürece dönmesi gereken bilgi belirtilerek, aşağıda açıklanmıştır.

AL

AL (*yerel–bağlandı–ismi*, *tampon–adresi*, *bayt–sayısı*)
-> *bayt–sayısı*, *aciliyet–bayrağı*, *gitsin–bayrağı*

Bu komut belirtilen bağlantı ile ilişkili bir tampon ayırır. Bu komuttan önce bir **AÇ** komutu yoksa veya çağıran süreç bu bağlantıyı kullanmaya yetkili değilse bir hata döner.

En basik gerçeklenimde, tampon dolmadıkça veya bir hata oluşmadıkça, denetim çağıran sürece dönmez fakat bu şema kısırdöngülere çok açıktır. Daha karmaşık gerçeklenimler bir kerede birden fazla AL çağrısına izin verirler. Bu tamponlar veribölütleri geldikçe doldurulur. Bu strateji bir GİTsin görüldüğünde veya bir tampon dolduğunda çağıran süreci uyaracak daha itinalı bir şema (muhtemelen eşzamansız) fiyatına işlenecek miktarı arttırmayı mümkün kılar.

Bir GİTsin görünmeden önce tamponu dolduracak yeterli veri gelirse GİTsin bayrağı **AL** çağrısının yanıtında etkin yapılmaz. Eğer GİTsin bayrağı tampon dolmadan önce görünürse tampon kısmen dolu döndürülür ve *gitsin–bayrağı* etkin kılınır.

Eğer acil veri varsa kullanıcı bir TCP'den kullanıcıya sinyal geliyormuş gibi uyarılacaktır. Alan kullanıcı bu nedenle "acil kip"te olmalıdır. Eğer *aciliyet–bayrağı* etkinse, ek olarak acil veri kalır. Eğer *aciliyet–bayrağı* etkin değilse, böyle bir **AL** çağrısı tüm acil veriyi döndürür, böylece kullanıcı "acil kip"i artık bırakabilir. Aciliyet göstericisinden sonra gelen veri (acil olmayan veri) önceki acil verinin sınırları kullanıcı için açıkça imlenmiş olmadıkça kullanıcıya önceki acil veriyle aynı tamponda teslim edilemez.

Tamponun tamamen dolmadığı durumu dikkate almak ve çözümlenmemiş **AL** çağrıları arasında ayrım yapabilmek için dönen kodun *tampon–göstericisi* ile alınan verinin asıl uzunluğunu belirten *bayt–sayısı* birlikte ele alınır.

Diğer bir **AL** gerçekleniminde ise tampon alanını TCP'nin ayırmasını gerektirebilir veya TCP bir döner tamponu kullanıcı ile paylaşabilir.

KAPAT

KAPAT (*yerel–bağlantı–ismi*)

Bu komut bağlantının KAPALI duruma geçirileceğini belirtir. Eğer bağlantı açık değilse veya çağıran süreç bağlantıyı kullanmaya yetkili değilse bir hata döner. Veriler gönderildikten sonra bir kapama komutu gönderilmesi, tamamlanmamış **GÖNDER**imlerin aktarılmasını (veya tekrardan aktarılmasını) sağlamak için kibarca bir uyarı görevini görür. Bu yüzden bir kaç tane **GÖNDER** komutunun verilmesinin ardından bir **KAPAT** komutu vermekle bütün verilerin hedef adrese iletildiği kanısına varmamız kabul edilebilir bir düşünce olur. KAPANIŞ durumundaki bir bağlantı üzerinde kullanıcılar hala veri **AL**maya devam ediyor olabilirler, çünkü karşı taraf verisinin son kısmını göndermeyi bu durumda iken bile deniyor olabilir. Buradan da anlıyoruz ki **KAPAT** demekle "artık veri almıyacağım" değil, "artık gönderecek bir verim yok" demiş oluyoruz. Şöyle bir şey olabilir (kullanıcı seviyesi protokol iyi tasarlanmamışsa): **KAPAT** komutunu veren taraf bütün verilerinden zamanaşımından önce kurtulamayabilir. Bu olayda, **KAPAT**, **TERKET**'e dönüşür ve kapatan TCP işlemi terkeder.

Kullanıcı bağlantıyı herhangi bir anda kendi inisiyatifinde **KAPAT**abileceği gibi TCP'den gelen çeşitli teşviklere (uzaktan kapatma, aktarımda zamanaşımı, hedefin erişilebilir olmayışı gibi) yanıt olarak da **KAPAT**abilir.

Bir bağlantının kapanması yabancı TCP ile iletişimi gerektirdiğinden bağlantılar kısa bir süre için KAPANIŞ durumunda kalabilirler. **KAPAT** komutu TCP tarafından yanıtlanmadan önce bağlantının yeniden açılmaya çalışılması, hata yanıtlarıyla sonuçlanacaktır.

Kapatma işlemi ayrıca gitsin işlemine de yol açar.

DURUM

DURUM (*yerel–bağlantı–ismi*) -> *durum–verisi*

Bu gerçeklenim bağımlısı bir kullanıcı komutu olup olmayışının olumsuz bir etkisi olmayacaktır. Dönen bilgi genellikle bağlantı ile ilişkili ADB'den gelir.

Bu komut şu bilgileri içeren bir veri bloku ile döner:

```
yerel soket,
yabancı soket,
yerel bağlantı ismi,
alım penceresi,
gönderi penceresi,
bağlantı durumu,
alındı bekleyen tampon sayısı,
alınmayı bekleyen tampon sayısı,
aciliyet durumu,
öncelik,
güvenlik/bölüm,
ve aktarım zamanaşımı.
```

Aktarımın durumuna veya gerçeklenimin kendisine bağlı olarak bu bilgilerin bir kısmı mevcut veya anlamlı olmayabilir. Eğer çağrıyı yapan süreç bağlantıyı kullanmaya yetkili değilse bir hata döner. Bu, yetkisiz bir sürecin bir bağlantı hakkında bilgi edinmeye çalışmasını önler.

TERKET

TERKET (*yerel–bağlantı–ismi*)

Bu komut, askıdaki tüm **GÖNDER** ve **AL**ların terkedilmesine, ADB'nin silinmesine ve bağlantının diğer tarafındaki TCP'ye özel bir baştan–başla iletisi gönderilmesine sebep olur. Gerçeklenime bağlı olarak, kullanıcılar yapılan her **GÖNDER** veya **AL** ile terketme istekleri alabilecekleri gibi basitçe **TERKET**'li alındılar da alabilirler.

3.8.1.2. TCP'nin Kullanıcıya İletileri

İşletim sisteminin TCP'ye kullanıcı programını eşzamansız olarak sinyalleyebilme ortamını sağladığı kabul edilir. TCP kullanıcı programını sinyallerken kullanıcıya bazı bilgiler aktarılılır. Belirtimde çoğunlukla bilgi bir hata iletisi olacaktır. Diğer durumlarda, bir **GÖNDER** veya **AL** ya da başka bir kullanıcı çağrısının işini tamamlamasıyla ilgili bilgiler varolacaktır.

Şu bilgiler sağlanır:

Yerel Bağlantı İsmi	Daima
Yanıt Dizgesi	Daima
Tampon Adresi	Gönderi ve Alım
Bayt Sayısı (alınan)	Alım
GİTsin Bayrağı	Alım
ACiL bayrağı	Alım

3.8.2. TCP/Düşük–Seviye Arayüzü

TCP çağrıları ağ üzerinden bilgileri aslında bir düşük seviyeli protokol üzerinden alır ve gönderir. ARPA ağlararası sistemindeki düşük seviyeli modül için tek seçenek Genel Ağ Protokolüdür (IP) [2].

Düşük seviyeli protokol IP ise argümanlar bir hizmet türü için bir yaşam süresince sağlanır. TCP bu parametreler için şu ayarları kullanır:

Hizmet Türü= Öncelik: sıradan, Gecikme: normal, İşlenen miktar: normal, Güvenirlik: normal; veya 00000000.
Yaşam Süresi= bir dakika veya 00111100.

> Varsayılan Azami Veribölütü Ömrü'nün (AVÖ) iki dakika olduğuna dikkat ediniz. Burada biz açıkça bir veribölütünün Genel Ağ sisteminde bir dakika içinde teslim edilememesi halinde yokedilmesini istemiş oluyoruz.

Eğer düşük seviyeli protokol IP (veya aynı özellikteki başka bir protokol) ise ve kaynak yönlendirmesi kullanılmışsa, arayüz yönlendirme bilgisinin iletişimine izin vermelidir. TCP sağlama özetinde kullanılan kaynak ve hedef adreslerinin başlatan kaynak ve nihai hedef olmasından dolayı bu özellikle önemlidir. Bağlantı isteklerine yanıt olarak dönen rotanın korunması da ayrıca önemlidir.

Bir düşük seviyeli protokol, gerek IP'ye işlevsel olarak eşdeğer hizmeti sağlamak, gerekse TCP sağlama özetinde kullanmak için kaynak adresini, hedef adresini, protokol alanlarını ve "TCP uzunluğu"nun saptamanın bir yolunu sağlamak zorundadır.

3.9. Olay İşleme

Bu bölümde betimlenen işlemler olası bir gerçeklenim örneğidir. Başka gerçeklenimlerin işlem sıraları birazcık farklı olabilir, fakat bu bölümdekilerden esasta değil sadece ayrıntıda farklı olmalıdırlar.

TCP etkinliğinin ayırıcı özellikleri olaylara verdiği yanıtlardır denebilir. Meydana gelen olaylar üç kategoride incelenebilir: kullanıcı çağrıları, veribölütlerinin varışı ve zamanaşımları. Bu bölümde her bir olaya TCP'nin verdiği işlemsel yanıtlar açıklanmıştır. Çoğu durumda gereken işlem bağlantının durumuna bağımlıdır.

Meydana gelen olaylar:

Kullanıcı Çağrıları

```
AÇ
GÖNDER
AL
KAPAT
TERKET
DURUM
```

Ulaşan Veribölütleri

```
Veribölütü Varışları
```

Zamanaşımları

```
Kullanıcı Zamanaşımı
Yeniden Aktarım Zamanaşımı
ZMN-BEKLE Zamanaşımı
```

TCP/Kullanıcı arayüzü modeli hemen bir dönüş ve bir olay veya bir sözde kesme üzerinden olası bir gecikmiş yanıt alan kullanıcı komutlarından oluşur. Aşağıdaki açıklamalarda "sinyal" bir gecikmiş yanıta sebep olan şey anlamında kullanılmıştır.

Hata yanıtları karakter dizgeleri olarak verilmiştir. Örneğin, mevcut olmayan bağlantılara atıf yapan kullanıcı komutları şu yanıtı alır: "hata: bağlantı açık değil".

Sıra numaraları, alındı numaraları, pencereler, vesaire ile ilgili tüm aritmetik işlemler 2^32'lik bir sıra numaraları uzayıyla sınırlıdır. Ayrıca "=<" işareti 2^32 ile bölümden artandan küçük veya eşit oduğunu gösterir.

Gelen veribölütlerine uygulanan işlemlerde izlenen doğal yol, önce sıra numarasının doğruluğunun sınanması (sıra numarası uzayının beklenen "alım penceresi" aralığına düşen sıra numaralarından biri olup olmadığı) ve bu sıra numarasına göre kuyruğa alınıp işlenmesidir.

Bir veribölütü daha önce alınmış veribölütleri ile örtüştüğünde, veribölütünü sadece yeni veriyi içerecek şekilde yeniden oluşturur ve başlık alanlarını uygun biçimde ayarlarız.

Bir durum değişikliğinden bahsedilmedikçe TCP'nin aynı durumda kalacağına dikkat ediniz.

3.9.1. AÇ Çağrısı

KAPALI Durumu (örn, ADB'nin olmayışı)

Bağlantı durum bilgisini saklamak üzere yeni bir Aktarım Denetim Bloku (ADB) oluşturulur. Yerel soket belirteci, yabancı soket, öncelik, güvenlik/bölüm ve kullanıcı zamanaşımı bilgileri doldurulur. Yabancı soketin bazı parçalarının bir edilgen **AÇ** çağrısında belirsiz olabileceğini ve gelen EŞZ veribölütündeki parametrelere göre doldurulacağına dikkat ediniz. Bu kullanıcı için istenen güvenlik ve önceliğin doğrulanmasına izin verilebilmesi için "hata: önceliğe izin verilmiyor" veya "hata: güvenlik/bölüme izin verilmiyor" şeklinde bir hata dönmemiş olması gerekir. Eğer çağrı edilgen ise DİNLE durumuna geçilir ve dönülür. Çağrı etkin ise ve yabancı soket belirsizse, "hata: yabancı soket belirsiz" hatası döner. Çağrı etkin ise ve yabancı soket belirliyse, bir EŞZ veribölütü hazırlanır. Bir ilk gönderi sıra nunarası (İGS) seçilip <SIRA=İGS><DNT=EŞZ> biçiminde bir EŞZ veribölütü gönderilir. GÖN.OLM değişkenine İGS, GÖN.SNR değişkenine İGS+1 atanır, EŞZ-GÖNDER durumuna geçilir ve döner.

Eğer çağrıcı belirtilen yerel sokete erişemiyorsa, "hata: bağlantı bu süreç için kuraldışı" hatası döner. Yeni bir bağlantı oluşturmak için yeterli yer yoksa, "hata: özkaynaklar yetersiz" hatası döner.

DİNLE Durumu

Çağrı etkin ve yabancı soket belirli ise, bağlantı edilgenden etkine döner, bir İGS seçilir. Bir EŞZ veribölütü gönderilir, GÖN.OLM değişkenine İGS, GÖN.SNR değişkenine İGS+1 atanır. EŞZ-GÖNDER durumuna geçilir. **GÖNDER** ile ilişkili veri EŞZ veribölütü ile gönderileceği gibi KURULU duruma geçildikten sonraki aktarım için kuyruğa da alınabilir. Aciliyet bitinin veri ile gönderilmesi komutta istenmişse, veribölütleri bu komutun bir sonucu olarak gönderilir. İstek için kuyrukta yer yoksa, yanıt "hata: özkaynaklar yetersiz" olur. Eğer yabancı soket belirtilmemişse, "hata: yabancı soket belirsiz" hatası dönülür.

EŞZ-GÖNDER Durumu
EŞZ-ALINDI Durumu
KURULU Durumu
SON-BEKLE-1 Durumu
SON-BEKLE-2 Durumu
KAPAT-BEKLE Durumu
KAPANIŞ Durumu
SON-ALN Durumu
ZMN-BEKLE Durumu

"hata: bağlantı zaten mevcut" hatası döner.

3.9.2. GÖNDER Çağrısı

KAPALI Durumu (örn, ADB'nin olmayışı)

Eğer kullanıcının böyle bir bağlantıya erişim izni yoksa, "hata: bağlantı bu süreç için kuraldışı" hatası döner.

Aksi takdirde, "hata: bağlantı yok" hatası döner.

DİNLE Durumu
Yabancı soket belirliyse, bağlantı edilgenden etkine döner, bir İGS seçilir. Bir EŞZ veribölütü gönderilir, GÖN.OLM değişkenine İGS, GÖN.SNR değişkenine İGS+1 atanır. EŞZ-GÖNDER durumuna geçilir. **GÖNDER** ile ilişkili veri EŞZ veribölütü ile gönderileceği gibi KURULU duruma geçildikten sonraki aktarım için kuyruğa da alınabilir. Aciliyet bitinin veri ile gönderilmesi komutta istenmişse, veribölütleri bu komutun bir sonucu olarak gönderilir. İstek için kuyrukta yer yoksa, yanıt "hata: özkaynaklar yetersiz" olur. Eğer yabancı soket belirtilmemişse, "hata: yabancı soket belirsiz" hatası döner.

EŞZ-GÖNDER Durumu
EŞZ-ALINDI Durumu
KURULU duruma geçildikten sonra aktarılmak üzere veri kuyruğa alınır. Kuyrukta yer yoksa, yanıt "hata: özkaynaklar yetersiz" olur.

KURULU Durumu
KAPAT-BEKLE Durumu
Tampon veribölütlenir ve bir alındı bindirilip (alındı değeri= ALM.SNR) ile gönderilir. Bu tamponu hatırlamak için yeterli yer yoksa, yanıt "hata: özkaynaklar yetersiz" olur.

Aciliyet bayrağı etkinse, GÖN.ACL <- GÖN.SNR-1 yapılır ve giden veribölütlerinde aciliyet göstericisi etkinleştirilir.

SON-BEKLE-1 Durumu
SON-BEKLE-2 Durumu
KAPANIŞ Durumu
SON-ALN Durumu
ZMN-BEKLE Durumu
"hata: bağlantı kapanıyor" döner ve istek yerine getirilmez.

3.9.3. AL Çağrısı

KAPALI Durumu (örn, ADB'nin olmayışı)
Eğer kullanıcının böyle bir bağlantıya erişim izni yoksa, "hata: bağlantı bu süreç için kuraldışı" hatası döner.

Aksi takdirde, "hata: bağlantı yok" hatası döner.

DİNLE Durumu
EŞZ-GÖNDER Durumu
EŞZ-ALINDI Durumu
KURULU duruma geçildikten sonra aktarılmak üzere veri kuyruğa alınır. Kuyrukta yer yoksa, yanıt "hata: özkaynaklar yetersiz" olur.

KURULU Durumu
SON-BEKLE-1 Durumu
SON-BEKLE-2 Durumu
İsteği yerine getirmek için gelen veribölütlerinden kuyruğa alınanlar yetersizse, istek kuyruğa alınır. **AL** çağrısını hatırlamak için kuyruk alanı yoksa, yanıt "hata: özkaynaklar yetersiz" olur.

Kuyruktaki gelen veribölütleri alım tamponunda yeniden oluşturulur ve kullanıcıya dönülür. Durum uygunsa, GİTsin imlemesi yapılır.

ALM.ACL kullanıcıya aktarılmakta olan verinin önündeyse kullanıcı acil veri varlığı konusunda uyarılır.

TCP, verinin kullanıcıya tesliminden sorumlu olduğunda, kullanıcıyla iletişimin bir alındı üzerinden yapılması gerekir. Böyle bir alındının oluşumu, aşağıda, bir gelen veribölütünün işlenmesi konusunda ele alınacaktır.

KAPAT-BEKLE Durumu
Uzak taraf zaten bir SON göndermiş olduğundan, **AL** çağrıları elde mevcut olup da henüz kullanıcıya teslim edilmemiş metinden oluşmalıdır. Teslimatı bekleyen metin yoksa, **AL** çağrısı "hata: bağlantı kapanıyor" yanıtını alacaktır. Aksi takdirde, bekleyen metin **AL** çağrısını oluşturmakta kullanılabilir.

KAPANIŞ Durumu
SON-ALN Durumu
ZMN-BEKLE Durumu
"hata: bağlantı kapanıyor" döner.

3.9.4. KAPAT Çağrısı

KAPALI Durumu (örn, ADB'nin olmayışı)
Eğer kullanıcının böyle bir bağlantıya erişim izni yoksa, "hata: bağlantı bu süreç için kuraldışı" hatası döner.

Aksi takdirde, "hata: bağlantı yok" hatası döner.

DİNLE Durumu
Askıdaki **AL** çağrıları "hata: kapanış" yanıtları ile döndürülür. ADB silinir. KAPALI durumu geçilir ve dönülür.

EŞZ-GÖNDER Durumu
Kuyruktaki **GÖNDER** veya **AL** çağrıları "hata: kapanış" yanıtları ile döndürülür. ADB silinir.

EŞZ-ALINDI Durumu
Hiç **GÖNDER** yoksa ve gönderilecek bekleyen veri de yoksa, bir SON veribölütü oluşturulup o gönderilir ve SON-BEKLE-1 durumuna geçilir; aksi takdirde, KURULU duruma geçince işlenmek üzere kuyruğa alınır.

KURULU Durumu
Veribölütlerine bölünerek **GÖNDER** çağrıları haline getirilmiş verilerin tümü kuyruğa alındıktan sonra bir SON veribölütü oluşturulur ve o gönderilir. Her durumda, SON-BEKLE-1 durumuna geçilir.

SON-BEKLE-1 Durumu
SON-BEKLE-2 Durumu
Kesinlikle bu bir hatadır ve bir "hata: bağlantı kapanıyor" yanıtı alınmalıdır. İkinci bir SON yayınlanana kadar (ilk SON yeniden aktarılabilirse de) bir "tamam" da kabul edilebilir bir yanıt olurdu.

KAPAT-BEKLE Durumu
Önceki tüm **GÖNDER** çağrıları veribölütleri haline getirilene kadar bu istek kuyruğa alınır; sonra bir SON veribölütü gönderilir, KAPANIŞ durumuna geçilir.

KAPANIŞ Durumu
SON-ALN Durumu
ZMN-BEKLE Durumu
"hata: bağlantı kapanıyor" yanıtı alınır.

3.9.5. TERKET Çağrısı

KAPALI Durumu (örn, ADB'nin olmayışı)

Eğer kullanıcının böyle bir bağlantıya erişim izni yoksa, "hata: bağlantı bu süreç için kuraldışı" hatası döner.

Aksi takdirde, "hata: bağlantı yok" hatası döner.

DİNLE Durumu
Askıdaki **AL** çağrıları "hata: bağlantı baştan başlatılıyor" yanıtları ile döndürülür. ADB silinir. KAPALI durumu geçilir ve dönülür.

EŞZ-GÖNDER Durumu
Kuyruktaki **GÖNDER** veya **AL** çağrıları "hata: bağlantı baştan başlatılıyor" yanıtları ile döndürülür. ADB silinir. KAPALI durumu geçilir ve dönülür.

EŞZ-ALINDI Durumu
KURULU Durumu
SON-BEKLE-1 Durumu
SON-BEKLE-2 Durumu
KAPAT-BEKLE Durumu
Bir BŞT veribölütü gönderilir.

```
<SIRA=GÖN.SNR><DNT=BŞT>
```

Kuyruktaki tüm **GÖNDER** veya **AL** çağrıları "hata: bağlantı baştan başlatılıyor" yanıtları ile döndürülmeli; aktarım veya yeniden aktarım için kuyruğa alınmış tüm veribölütleri (yukarıda BŞT için biçimlenmiş olan hariç) boşaltılmalı, ADB silinmeli, KAPALI duruma geçilip dönülmelidir.

KAPANIŞ Durumu
SON-ALN Durumu
ZMN-BEKLE Durumu
"tamam" yanıtı verilip ADB silinir, KAPALI duruma geçilip dönülür.

3.9.6. DURUM Çağrısı

KAPALI Durumu (örn, ADB'nin olmayışı)
Eğer kullanıcının böyle bir bağlantıya erişim izni yoksa, "hata: bağlantı bu süreç için kuraldışı" hatası döner.

Aksi takdirde, "hata: bağlantı yok" hatası döner.

DİNLE Durumu
"durum= DİNLE" ve ADB göstericisi döner.

EŞZ-GÖNDER Durumu
"durum= EŞZ–GÖNDER" ve ADB göstericisi döner.

EŞZ-ALINDI Durumu
"durum= EŞZ–ALINDI" ve ADB göstericisi döner.

KURULU Durumu
"durum= KURULU" ve ADB göstericisi döner.

SON-BEKLE-1 Durumu
"durum= SON–BEKLE–1" ve ADB göstericisi döner.

SON-BEKLE-2 Durumu
"durum= SON–BEKLE–2" ve ADB göstericisi döner.

KAPAT-BEKLE Durumu

"durum= KAPAT–BEKLE" ve ADB göstericisi döner.

KAPANIŞ Durumu
"durum= KAPANIŞ" ve ADB göstericisi döner.

SON–ALN Durumu
"durum= SON–ALN" ve ADB göstericisi döner.

ZMN–BEKLE Durumu
"durum= ZMN–BEKLE" ve ADB göstericisi döner.

3.9.7. Veribölütü Varışları

Durum, **KAPALI** ise (örn, ADB'nin olmayışı)
Gelen veribölütündeki tüm veri iptal edilir. BŞT içeren veribölütü varsa iptal edilir. Bir BŞT içermeyen bir gelen veribölütü yanıt olarak bir BŞT gönderilmesine sebep olur. Alındı ve sıra numarası alanlarının değerleri suçlu veribölütünü gönderen TCP tarafından baştan başlatma işlemi için kabul edilebilir şekilde seçilir.

ALN biti etkin değilse, sıra numarası olarak sıfır kullanılır.

```
<SIRA=0><ALN=VBL.SIRA+VBL.UZN><DNT=BŞT,ALN>
```

ALN biti etkinse,

```
<SIRA=VBL.ALN><DNT=BŞT>
```

Ve döner.

Durum, **DİNLE** ise
İlk sınama bir BŞT için yapılır.

Gelen BŞT yoksayılmalıdır. Dönülür.

İkinci sınama bir ALN için yapılır.

Bir bağlantı hala DİNLE durumundayken bir alındı gelirse kötüdür. ALN kılıklı bir veribölütü gelmişse bir kabul edilebilir BŞT veribölütü oluşturulur:

```
<SIRA=VBL.ALN><DNT=BŞT>
```

Dönülür.

Üçüncü sınama bir EŞZ için yapılır.

EŞZ biti etkinse, güvenlik sınanır. Eğer gelen veribölütündeki güvenlik/bölüm ile ADB'deki tam olarak uyuşmuyorsa bir BŞT veribölütü gönderilir ve dönülür.

```
<SIRA=VBL.ALN><DNT=BŞT>
```

VBL.PRC > ADB.PRC ise, kullanıcı ve sistem tarafından izin verilmişse ADB.PRC<-VBL.PRC atanır; izin verilmemişse, bir BŞT veribölütü gönderilir ve dönülür.

```
<SIRA=VBL.ALN><DNT=BŞT>
```

VBL.PRC < ADB.PRC ise devam edilir.

ALM.SNR değişkenine VBL.SIRA+1 atanır, İAS değişkenine VBL.SIRA atanır ve başka her denetim ve metin daha sonra işlenmek üzere kuyruğa alınır. İGS seçilmeli ve şöyle bir EŞZ veribölütü gönderilmelidir:

```
<SIRA=İGS><ALN=ALM.SNR><DNT=EŞZ,ALN>
```

GÖN.SNR değişkenine İGS+1 ve GÖN.OLM değişkenine İGS atanır. Bağlantı durumu EŞZ-ALINDI yapılır. Gelen başka her denetim ve veri (EŞZ ile birleşik) EŞZ-ALINDI durumunda işlenecektir, fakat EŞZ ve ALN tekrar işleme sokulmaz. Dinleme tamamen belli değilse (örn, yabancı sotetin tamamen belli olmaması durumu), belirsiz alanlar şimdiden doldurulmalıdır.

Dördüncü sınama başka metin veya denetim için yapılır.

Bir başka denetim veya metin kılıklı veribölütü (EŞZ içermeyen) bir ALN içermelidir; böylece ALN işlemi tarafından iptal edilir. Bağlantının bu varoluşu tarafından gönderilmiş hiçbir şeyin yanıtı olamayacağından bir gelen BŞT veribölütü geçerli olamazdı. Bu yüzden burada böyle bir veribölütü almazsınız ama alırsanız da veribölütü iptal edilir ve dönülür.

Durum, **EŞZ-GÖNDER** ise

İlk sınama ALN biti için yapılır.

ALN biti etkinse

VBL.ALN =< İGS veya VBL.ALN > GÖN.SNR ise, bir BŞT veribölütü gönderilir (BŞT biti etkin olmadıkça; aksi takdirde, veribölütü iptal edilip dönülür).

```
<SIRA=VBL.ALN><DNT=BŞT>
```

ve veribölütü iptal edilir. Dönülür.

GÖN.OLM =< VBL.ALN =< GÖN.SNR ise ALN kabul edilebilirdir.

İkinci sınama BŞT biti için yapılır

BŞT biti etkinse

ALN kabul edilebilir ise kullanıcı "hata: bağlantı baştan başlatılıyor" sinyalini alır, veribölütü iptal edilir, KAPALI duruma geçilir, ADB silinir ve dönülür. Aksi takdirde (ALN yoksa), veribölütü iptal edilir ve dönülür.

Üçüncü sınama güvenlik ve öncelik için yapılır.

Eğer veribölütündeki güvenlik/bölüm ile ADB'deki tam olarak uyuşmuyorsa bir BŞT veribölütü gönderilir.

Bir ALN varsa

```
<SIRA=VBL.ALN><DNT=BŞT>
```

Aksi takdirde

```
<SIRA=0><ALN=VBL.SIRA+VBL.UZN><DNT=BŞT,ALN>
```

Bir ALN varsa

Eğer veribölütündeki öncelik ile ADB'deki tam olarak uyuşmuyorsa bir BŞT veribölütü gönderilir.

```
<SIRA=VBL.ALN><DNT=BŞT>
```

Bir ALN yoksa

Eğer veribölütündeki öncelik ADB'dekinden daha yüksekse, kullanıcı ve sistem tarafından izin verilmişse, ADB'deki öncelik veribölütündekine yükseltilir; önceliğin yükseltilmesine izin verilmezse, bir BŞT veribölütü gönderilir.

```
<SIRA=0><ALN=VBL.SIRA+VBL.UZN><DNT=BŞT,ALN>
```

Eğer veribölütündeki öncelik ile ADB'dekinden daha düşükse devam edilir.

Eğer bir BŞT veribölütü gönderilmişse veribölütü iptal edilir ve dönülür.

Dördüncü sınama EŞZ biti için yapılır

Bu adıma sadece ALN tamamsa geçilmelidir, değilse bir ALN yoktur ve veribölütü bir BŞT içermez.

EŞZ biti etkinse ve güvenlik/bölüm ve öncelik kabul edilebilir ise, ALM.SNR = VBL.SIRA+1 ve İAS = VBL.SIRA yapılır. GÖN.OLM değeri VBL.ALN'ye eşit olacak şekilde arttırılmalı (bir ALN varsa) ve yeniden aktarım kuyruğundaki veribölütleri alındılandıklarından silinmelidir.

GÖN.OLM > İGS (bizim EŞZ, ALN'lenmişti) ise, bağlantı durumu KURULU yapılır ve bir ALN veribölütü oluşturulur ve gönderilir:

<SIRA=GÖN.SNR><ALN=ALM.SNR><DNT=ALN>

Aktarım için kuyruğa alınmış veri ve denetimler de dahil edilebilir. Veribölütünde başka veri ve denetim yoksa ACL bitinin sınandığı altıncı adımda işleme devam edilir, aksi takdirde dönülür.

Aksi takdirde, EŞZ-ALINDI durumuna geçilir, bir EŞZ,ALN veribölütü oluşturulur ve gönderilir:

<SIRA=İGS><ALN=ALM.SNR><DNT=EŞZ,ALN>

Veribölütünde başka veri ve denetim varsa, KURULU duruma geçildikten sonra işlenmek üzere kuyruğa alınır ve dönülür.

Beşinci sınamada, ne EŞZ ne de BŞT biti etkinse, veribölütü iptal edilip dönülür.

Aksi takdirde,

İlk sınama sıra numarası için yapılır

EŞZ-ALINDI Durumu

KURULU Durumu

SON-BEKLE-1 Durumu

SON-BEKLE-2 Durumu

KAPAT-BEKLE Durumu

KAPANIŞ Durumu

SON-ALN Durumu

ZMN-BEKLE Durumu

Veribölütleri sırayla işlenir. Ulaşanlar üzerindeki ilk denemeler eski yinelenmişleri iptal etmek içindir, fakat bu işlem VBL.SIRA sırasıyla yapılır. Bir veribölütünün içeriği eski ve yeniyi birlikte içeriyorsa, sadece yeni parçalar işlenmelidir.

Gelen bir veribölütünün kabul edilebilirliği için dört durum sözkonusudur:

Veribölütü Uzunluğu	Alım Penceresi	Sınama
0	0	VBL.SIRA = ALM.SNR
0	>0	ALM.SNR =< VBL.SIRA < ALM.SNR+ALM.PEN
>0	0	kabul edilebilir değil
>0	>0	ALM.SNR =< VBL.SIRA < ALM.SNR+ALM.PEN veya ALM.SNR =< VBL.SIRA+VBL.UZN-1 < ALM.SNR+ALM.PEN

ALM.PEN sıfırsa, hiçbir veribölütü kabul edilmez, fakat geçerli ALN, ACL ve BŞT'leri kabul etmek için özel bir izin verilmelidir.

Gelen bir veribölütü kabul edilebilir değilse, yanıtta bir alındı gönderilmelidir (BŞT biti etkin olmadıkça; etkinse veribölütü iptal edilip dönülür).

<SIRA=GÖN.SNR><ALN=ALM.SNR><DNT=ALN>

Bir alındı gönderildikten sonra kabul edilmeyen veribölütü iptal edilir ve dönülür.

Onu izleyen veribölütünün ALM.SNR'de başlayan ve pencereyi aşmayan idealleştirilmiş veribölütü olduğu varsayılır. Asıl veribölütleri bu kabule sığması için pencerenin (EŞZ ve SON dahil) dışına düşen kısımları kırpılarak ve sadece ALM.SNR'de başlayan veribölütü işlenerek yeniden biçimlendirilebilir. Daha yüksek sira numaraları ile başlayan veribölütleri daha sonra işlenmek üzere tutulabilir.

İkinci sınama BŞT biti için yapılır.

EŞZ-ALINDI Durumu

BŞT biti etkinse

Bağlantı bir edilgen **AÇ** çağrısı ile başlatılmışsa (örn, DİNLE durumundan gelinip), bağlantı DİNLE durumuna geçirilir ve dönülür. Kullanıcıya bilgi verilmesine gerek yoktur. Bağlantı bir etkin **AÇ** çağrısı ile başlatılmışsa (örn, EŞZ-GÖNDER durumundan gelinip), bağlantı reddedilir ve kullanıcıya "bağlantı reddedildi" sinyali yollanır. Her durumda, yeniden aktarım kuyruğundaki tüm veribölütleri silinir. Ve etkin **AÇ** durumunda, KAPALI duruma geçilip ADB silinir ve dönülür.

KURULU

SON-BEKLE-1

SON-BEKLE-2

KAPAT-BEKLE

BŞT biti etkinse askıdaki **AL** ve **GÖNDER** çağrıları "baştan–başlat" yanıtları almalıdır. Tüm veribölütü kuyrukları boşaltılır. Kullanıcılar ayrıca talep edilmemiş bir genel "bağlantı yeniden başlatılıyor" sinyali almalıdır. KAPALI duruma geçildikten sonra ADB silinir ve dönülür.

KAPANIŞ Durumu

SON-ALN Durumu

ZMN-BEKLE

BŞT biti etkinse KAPALI duruma geçildikten sonra ADB bilinir ve dönülür.

Üçüncü sınama güvenlik ve öncelik için yapılır.

EŞZ-ALINDI Durumu

Eğer veribölütündeki güvenlik/bölüm ve öncelik ile ADB'dekiler tam olarak uyuşmuyorsa bir BŞT veribölütü gönderilir ve dönülür.

KURULU Durumu

Eğer veribölütündeki güvenlik/bölüm ve öncelik ile ADB'dekiler tam olarak uyuşmuyorsa bir BŞT veribölütü gönderilir ve askıdaki **AL** ve **GÖNDER** çağrıları "yeniden–başlat" yanıtları alırlar. Tüm veribölütü kuyrukları boşaltılır. Kullanıcılar ayrıca talep edilmemiş bir genel "bağlantı yeniden başlatılıyor" sinyali almalıdır. KAPALI duruma geçildikten sonra ADB silinir ve dönülür.

Bu portlar arasındaki farklı güvenlik veya öncelikli eski bir bağlantıdaki bir veribölütünün mevcut bağlantının terkedilmesine sebep olmadan engellenmesi için bu sınamanın aşağıdaki sıra numarası sınamasında yeraldığına dikkat ediniz.

Dördüncü sınama EŞZ biti için yapılır.

EŞZ-ALINDI Durumu

KURULU Durumu

SON-BEKLE-1 Durumu

SON-BEKLE-2 Durumu

KAPAT-BEKLE Durumu

KAPANIŞ Durumu

SON-ALN Durumu

ZMN-BEKLE Durumu

EŞZ pencere içindeyse o bir hatadır, bir BŞT veribölütü gönderilir, askıdaki **AL** ve **GÖNDER** çağrıları "baştan–başlat" yanıtları alır, tüm veribölütü kuyrukları boşaltılır, kullanıcılar ayrıca talep edilmemiş bir genel "bağlantı yeniden başlatılıyor" sinyali alır, KAPALI duruma geçildikten sonra ADB silinir ve dönülür.

EŞZ pencere içinde değilse bu adıma gelinmeden, ilk adımda (sıra numarası sınaması) bir alındı gönderilirdi.

Beşinci sınama ALN alanı için yapılır.

ALN biti etkin değilse veribölütü iptal edilir ve dönülür

ALN biti etkinse

EŞZ-ALINDI Durumu

GÖN.OLM =< VBL.ALN =< GÖN.SNR ise KURULU duruma geçilip işlem sürdürülür.

Veribölütü alındısı kabul edilebilir değilse bir BŞT veribölütü oluşturulur:

```
<SIRA=VBL.ALN><DNT=BŞT>
```

ve gönderilir.

KURULU Durumu

GÖN.OLM < VBL.ALN =< GÖN.SNR ise GÖN.OLM <- VBL.ALN yapılır. Yeniden aktarım kuyruğundaki veribölütleri alındılandıklarından silinmelidir. Kullanıcılar **GÖNDER**ilmiş ve tamamen alındılanmış tamponlar için olumlu alındılar almalıdır (örn, **GÖNDER** tamponu "tamam" yanıtı ile dönmelidir). ALN yinelenmişse (VBL.ALN < GÖN.OLM), bu yoksayılır. Birşeylerin henüz gönderilmemiş alındıları varsa (VBL.ALN > GÖN.SNR) bir ALN gönderilir, veribölütü yokedilir ve dönülür.

GÖN.OLM < VBL.ALN =< GÖN.SNR ise, gönderi penceresi güncellenmemelidir. (GÖN.PS1 < VBL.SIRA veya (GÖN.PS1 = VBL.SIRA ve GÖN.PS2 =< VBL.ALN)) ise, GÖN.PEN <- VBL.PEN, GÖN.PS1 <- VBL.SIRA ve GÖN.PS2 <- VBL.ALN yapılır.

GÖN.PEN'in, GÖN.OLM'den bir mesafe belirttiğine, GÖN.PS1'in GÖN.PEN'i güncellemekte kullanılan son veribölütünün sıra numarasını kaydettiğine ve GÖN.PS2'nin GÖN.PEN'i güncellemekte kullanılan son veribölütünün alındı numarasını kaydettiğine dikkat ediniz. Buradaki sınama eski veribölütleri kullanılarak pencerenin güncellenmesini engeller.

SON-BEKLE-1 Durumu

KURULU durumdaki işlemlere ek olarak, bizim SON alındılandığı anda SON-BEKLE-2 durumuna geçilir ve işleme bu durumda devam edilir.

SON-BEKLE-2 Durumu

KURULU durumdaki işlemlere ek olarak, yeniden aktarım kuyruğu boşsa kullanıcının **KAPAT**'ı alındılanabilir ("tamam") fakat ADB silinmez.

KAPAT-BEKLE Durumu

KURULU durumdaki işlemlerin aynısı yapılır.

KAPANIŞ Durumu

KURULU durumdaki işlemlere ek olarak, bizim SON'u alındılayan ALN varsa ZMN-BEKLE durumuna geçilir, yoksa veribölütü yoksayılır.

SON-ALN Durumu

Bu durumda gelebilen tek şey bizim SON'un bir alındısıdır. Bizim SON alındılandığı anda ADB silinir, KAPALI duruma geçilir ve dönülür.

ZMN-BEKLE Durumu
Bu durumda gelebilen tek şey uzak SON'un bir yeniden aktarımıdır. O tek şey alınır ve 2 AVÖ'lük zamanaşımından sonra yeniden başlanır.

Altıncı sınama ACL biti için yapılır.
KURULU Durumu
SON-BEKLE-1 Durumu
SON-BEKLE-2 Durumu
ACL biti etkinse ALM.ACL <- azm(ALM.ACL,VBL.ACL) yapılır ve kullanıcıya tüketilen verinin önünde aciliyet göstericisi (ALM.ACL) varsa uzak tarafın acil veriye sahip olduğu sinyallenir. Eğer kullanıcı zaten sinyallenmişse (veya hala "acil kip"te ise), devam niteliğindeki bu acil veri için kullanıcı tekrar sinyallenmez.

KAPAT-BEKLE Durumu
KAPANIŞ Durumu
SON-ALN Durumu
ZMN-BEKLE
Uzak taraf tarafından bir SON alınmış olduğundan bu olmamalıdır. ACL yoksayılır.

Yedinci sınama veribölütü metni için yapılır.
KURULU Durumu
SON-BEKLE-1 Durumu
SON-BEKLE-2 Durumu
KURULU durumdayken, veribölütü metninin kullanıcının **AL** tamponlarına teslimi mümkündür. Gerek tampon dolana gerekse veribölütü boşalana kadar veribölütlerindeki metin tamponlara taşınabilir. Veribölütü boş ve bir GİTsin bayrağı taşıyorsa, kullanıcı bilgilendirilir, tampon döndüğünde bir GİTsin alınmış olur.

TCP, verinin kullanıcıya tesliminin sorumluluğunu aldığında, ayrıca, verinin alımını da alındılamalıdır.

TCP, verinin sorumluluğunu aldığı anda, ALM.SNR kabul edilen veri kadar arttırılır ve tamponun kullanılabilirliği bakımından ALM.PEN ayarlanır. ALM.SNR ve ALM.PEN toplamı azalmamalıdır.

Veri İletişimi (sayfa: 29) bölümündeki pencere yönetim önerilerini lütfen dikkate alınız.

Şöyle bir alındı gönderilir:
```
<SIRA=GÖN.SNR><ALN=ALM.SNR><DNT=ALN>
```
Bu alındı, mümkünse yersiz bir gecikmeye uğramasızın, aktarılmakta olan bir veribölütüne bindirilmiş olmalıdır.

KAPAT-BEKLE Durumu
KAPANIŞ Durumu
SON-ALN Durumu
ZMN-BEKLE Durumu
Uzak tarafta bir SON alınmış olduğundan bu olmamalıdır. Veribölütü metni yoksayılır.

Sekizinci sınama SON biti için yapılır.
Durum KAPALI, DİNLE veya EŞZ-GÖNDER ise, VBL.SIRA doğrulanamayacağından, SON işleme alınmaz; veribölütü yokedilir ve dönülür.

SON biti etkinse, kullanıcıya "bağlantı kapanıyor" denip askıdaki **AL** çağrılarına aynı ileti döndürülür, ALM.SNR, SON kadar arttırılır ve SON için bir alındı gönderilir. Kullanıcıya henüz teslim edilmemiş her veribölütü metni için SON'un GİTsin uygulayacağına dikkat ediniz.

KURULU Durumu

EŞZ-ALINDI Durumu
KAPAT-BEKLE durumuna geçilir.

SON-BEKLE-1 Durumu
Bizim SON ALNlenmişse (belki bu veribölütünde), ZMN-BEKLE durumuna geçilir, zaman–bekle zamanlayıcısı çalıştırılır, diğer zamanlayıcılar kapatılır; aksi takdirde, KAPANIŞ durumuna geçilir.

SON-BEKLE-2 Durumu
ZMN-BEKLE durumuna geçilir, zaman–bekle zamanlayıcısı çalıştırılır, diğer zamanlayıcılar kapatılır.

KAPAT-BEKLE Durumu
KAPAT-BEKLE durumunda kalınır.

KAPANIŞ Durumu
KAPANIŞ durumunda kalınır.

SON-ALN Durumu
SON-ALN durumunda kalınır.

ZMN-BEKLE Durumu
ZMN-BEKLE durumunda kalınır. 2 AVÖ'lük zaman–bekle zamanaşımı başlatılır.

ve dönülür.

3.9.8. Kullanıcı Zamanaşımı

Herhangi bir durumda, kullanıcı zamanaşımı dolarsa, tüm kuyruklar boşaltılır, kullanıcı genellikle "hata: kullanıcı zamanaşımından dolayı bağlantı terkedildi" ile ve askıdaki çağrılar için sinyallenir, ADB silinir, KAPALI duruma geçilir ve dönülür.

3.9.9. Yeniden Aktarım Zamanaşımı

Herhangi bir durumda, yeniden aktarım kuyruğundaki bir veribölütünde yeniden aktarım zamanaşımı dolarsa, veribölütü tekrar yeniden aktarım kuyruğunun başına gönderilir, yeniden aktarım zamanaşımı yeniden başlatılır ve dönülür.

3.9.10. ZMN-BEKLE zamanaşımı

Bir bağlantı üzerinde ZMN-BEKLE zamanaşımı dolarsa ADB silinir, KAPALI duruma geçilir ve dönülür.

A. Kullanılan Terimler ve Kısaltmalar

1822

BBN Report 1822, "The Specification of the Interconnection of a Host and an IMP" ("Bir konak ile bir IMP'nin birbiriyle bağlantısının belirtimi"). Bir konak ile ARPANET arasındaki arayüzün belirtimi.

aciliyet göstericisi

Sadece `ACL` biti etkin olduğunda anlamlı olan bir denetim alanı. Bu alan, gönderen kullanıcının acil çağrısı ile ilişkili veri sekizlisini belirten aciliyet göstericisinin değerini haber verir.

`ACL`

Bir sıra numarası işgal etmeyen, alan kullanıcının, aciliyet göstericisinde belirtilen değerden küçük sıra numaraları taşıyan tüketilecek bir veri olduğu sürece acilen işlem yapması için uyarılmasını gerektiğini belirten bir denetim biti.

ADB

Aktarım Denetim Bloğu. Bir bağlantının durumunun kaydedildiği veri yapısı.

ADB.ÖNC

Bağlantı önceliği

alım penceresi

Almaya istekli yerel (alıcı) TCP'nin sıra numaralarını ifade eder. Böylece, yerel TCP, makbul veriyi ve denetimi taşıyan veribölütlerinin `ALM.SNR` ile `ALM.SNR + ALM.PEN - 1` arasında yeralacağını varsayar. Bu aralığın dışında sıra numaralarına sahip veribölütleri yinelenmiş sayılarak iptal edilirler.

`ALM.ACL`

alım aciliyet göstericisi

`ALM.PEN`

alım perceresi

`ALM.SNR`

sonraki alım sıra numarası

`ALN`

Bir veribölütünün alındı alanının, önceki tüm sıra numaralarının alındığını belirtmesi nedeniyle bu veribölütünün göndericisinin almayı umduğu sonraki sıra numarasını içerdiğini belirten ve hiçbir sıra alanı işgal etmeyen denetim bitinin ismi.

ARPANET iletisi

ARPANET'teki bir IMP ile bir konak arasındaki aktarım birimi. Azami boyu yaklaşık 1012 sekizlidir (8096 bit).

ARPANET paketi

ARPANET'teki IMP'ler arasında dahili olarak kullanılan bir aktarım birimi. Azami boyu yaklaşık 126 sekizlidir (1008 bit).

AVÖ

Azami Veribölütü Ömrü. Bir TCP veribölütünün ağ üzerinde mevcut olabildiği süre. Keyfi 2 dakika olarak tanımlanmıştır..

bağlantı

Bir çift soketin varlığı ile belirlenen mantıksal bir iletişim güzergahı.

başlık

Bir iletinin, veribölütü, veridilimi, veri paketi veya veri blokunun başlangıcındaki denetim bilgisi.

Betim
Bir Genel Ağ Protokolü alanı. Bu betimleme değeri kullanıcı tarafından, bir verikatarının veridilimlerinin montajına yardımcı olması için atanır.

EŞZ
Bir bağlantının başlatılmasında kullanılan, sıra numaralarının başlangıcını belirten, gelen veribölütünde bir sıra numarası işgal eden bir denetim biti (eşzamanlama kısaltması olarak).

FTP
("File Transfer Protocol" kısaltması) Dosya aktarım protokolü.

GİT
Bir sıra numarası işgal etmeyen, veribölütünün alıcı tarafa hemen gitmesi gereken veriyi içerdiğini belirten bir denetim biti.

GÖN.ACL
gönderi aciliyet göstericisi

GÖN.OLM
sol sıra

GÖN.PEN
gönderi penceresi

GÖN.PS1
son pencere güncellemesinde veribölütü sıra numarası

GÖN.PS2
son pencere güncellemesinde veribölütü alındı numarası

GÖN.SNR
gönderi sırası

gönderi penceresi
Almaya istekli uzak (alıcı) TCP'nin sıra numaralarını ifade eder. Uzak (veriyi alan) TCP'deki veribölütleri içinde belirtilen pencere alanının değeridir. Yeni sıra numarası aralığı bir TCP tarafından GÖN.SNR ile GÖN.OLM + GÖN.PEN − 1 arasında seçilir. (Şüphesiz, sıra numaralarının yeniden aktarımlarının GÖN.OLM ile GÖN.SNR arasında olması beklenir.)

gönderi sırası
Yerel (gönderen) TCP'nin bağlantıda kullanacağı sonraki sıra numarası. İlki bir ilk sıra numarası eğrisinden (İSN) seçilir ve aktarılan sıralı denetimin veya verinin her sekizlisi için bir arttırılır.

Hedef Adres
Genelde ağ ve konak betimleyen hedef adresi.

Hizmet Türü
Bir Genel Ağ veridilimi için hizmet türünü belirten bir Genel Ağ Protokolü alanı.

İAS
İlk Gelen Sıra numarası. Bir bağlantıda alıcı tarafından kullanılmış ilk sıra numarası.

İGS
İlk Giden Sıra numarası. Bir bağlantıda gönderici tarafından kullanılmış ilk sıra numarası.

IMP
("Interface Message Processor" kısaltması). Arayüz İleti İşlemci, ARPANET'in paket anahtarlaması.

Genel Ağ adresi
Özellikle konak seviyesinde bir kaynak veya hedef adresi.

Genel Ağ verikatarı
(İng.: Internet datagram) Genel Ağ başlıklı daha yüksek seviyeli bir protokol ile bir Genel Ağ modülü arasında değiştokuş edilen verinin birimi.

Genel Ağ veridilimi
(İng.: Internet fragment) Genel Ağ başlıklı bir Genel Ağ verikatarının veri parçası.

IP
Genel Ağ Protokolü. ("Internet Protocol" kısaltması)

İSN
İlk Sıra Numarası. Bir bağlantıda kullanılmış ilk sıra numarası (gerek İAS gerekse İGS olarak). Saate göre sıralı/döngülü bir düzende seçilir.

Kaynak Adresi
Ağ veya konak belirten kaynak adresi

konak
Bir bilgisayar. İletişim ağı bakımından iletiler için bir kaynak ya da hedef olan bilgisayar.

lider
Bir ileti veya veri blokunun başlangıcındaki denetim bilgisi. Özellikle, ARPANET'te, konak/IMP arayüzünde bir ARPANET iletisi üstündeki denetim bilgisi.

modül
Bir protokol veya başka bir yordamın genellikle yazılım olarak bir gerçeklenimi.

paket
Mantıksal olarak eksiği de olabilen bir başlığı olan bir veri paketi. Verinin mantıksal paketinden ziyade fiziksel paketi için kullanılır.

port
Veri ile ilişkili bir sürecin girdi veya çıktı kanalını belirten bir soket bölümü.

RTP
"Real Time Protocol" kısaltması. Gerçek Zamanlı Protokol: Zamanında işlenmesi önemli olan bilgilerin iletişimi için kullanılan iki konak arası protokol.

Seçenekler
Uzunluğu bir kaç sekizlik olabilen çeşitli seçenekler içeren bir seçenek alanı. Seçenekler öncelikle sınama amaçları için kullanılır; örneğin, zaman damgalarının taşınması. Genel Ağ Protokolü ve TCP, her ikisi de seçenek alanları içerir.

sekizli
(İng.: octet) Sekiz bitlik bayt. (Baytların bazı sistemlerde 8 bit olmaması yüzünden sekiz bitlik baytı belirtmek için)

BŞT

Bir sıra numarası işgal etmeyen, bir etkileşime girmeksizin alıcının bağlantıyı silmesi gerektiğini belirten bir denetim biti (baştan–başlat). Alıcı gelen veribölütünün alındı ve sıra numarası alanlarına bakarak bir baştan başlatma komutunu devreye alıp almayacağını saptayabilir. Yanıtında bir `BŞT` olmasına yolaçacak `BŞT` içeren bir veribölütü alımı söz konusu değildir.

soket
Özellikle bir port tanıtı içeren bir adres. Bir Genel Ağ Adresi ile bir TCP portunun birleşiminden oluşur.

sol sıra
Alan TCP tarafından alındılanacak sonraki sıra numarasıdır (veya henüz alındılanmamış en küçük sıra numarasıdır) ve bazan gönderi penceresinin sol kenarı olarak da atıfta bulunulur.

`SON`
Bir sıra numarası işgal eden, göndericinin artık veri göndermeyeceğini veya işgal edilen sıra numarası uzayının denetleneceğini belirter bir denetim biti (sonlanış).

sonraki alım sıra numarası
Yerel TCP'nin almayı umduğu sonraki sıra numarası.

süreç
Çalışmakta olan bir program. TCP'nin veya bir iki konak arası protokolün bakış açısından, verinin kaynağı veya hedefi.

TCP
(Transmission Control Protocol kısaltması) Aktarım Denetim Protokolü: Ağlararası ortamlarda güvenilir iletişim için bir iki–konakarası protokol.

HT
Hizmet Türü, bir Genel Ağ Protokolü alanı.

`VBL.ALN`
veribölütü alındısı

`VBL.UZN`
veribölütü uzunluğu

`VBL.PRC`
veribölütü öncelik değeri

`VBL.SIRA`
veribölütü sırası

`VBL.ACL`
veribölütü aciliyet göstericisi alanı

`VBL.PEN`
veribölütü pencere alanı

veribölütü
(İng.: segment) Mantıksal bir veri birimi. Konu özelinde, bir TCP veribölütü bir TCP modül çifti arasında aktarılan veri birimidir.

veribölütü alındısı
Gelen veribölütünün alındı alanındaki sıra numarası.

veribölütü sırası

Gelen veribölütünün sıra alanındaki numara.

veribölütü uzunluğu
Sıralama alanını işgal eden bir denetimi de içererek bir veribölütü tarafından işgal edilmiş sıra numarası alanı miktarı.

veridilimi
(İng.: fragment) Bir mantıksal veri biriminin bir parçası; bir Genel Ağ veridilimi bir Genel Ağ verikatarının bir parçasıdır.

verikatarı
(İng.: datagram) Bir paket anahtarlamalı bilgisayar ağında gönderilen bir ileti.

yerel paket
Bir yerel ağın içinde aktarım birimi

B. Kaynakça

1. [COM–22]
   Paket Ağı Arailetişimi için bir Protokol — ***A Protocol for Packet Network Intercommunication*** — **V. Cerf ve R. Kahn** — IEEE Transactions on Communications, Vol. COM–22, No. 5, pp 637–648 — Mayıs 1974

2. [RFC791 (B28)]
   Genel Ağ Prokotolü – DARPA Internet Programı Protokol Belirtimi — ***Internet Protocol – DARPA Internet Program Protocol Specification*** — **J. Postel (editör)** — USC/Information Sciences Institute — Eylül 1981

3. [CNv2]
   Aktarım Prokotollerinde Bağlantı Yönetimi — ***Connection Management in Transport Protocols*** — **Y. Dalal ve C. Sunshine** — Computer Networks, Vol. 2, No. 6, pp. 454–473 — Aralık 1978

4. [RFC790]
   Atanmış Numaralar — ***Assigned Numbers*** — **J. Postel** — USC/Information Sciences Institute — Eylül 1981

Notlar

Belge içinde dipnotlar ve dış bağlantılar varsa, bunlarla ilgili bilgiler bulundukları sayfanın sonunda dipnot olarak verilmeyip, hepsi toplu olarak burada listelenmiş olacaktır.

(B1) ftp://ftp.rfc-«editor.org/in-«notes/bcp/bcp78.txt

(B3) http://www.ietf.org/

(1) Deputy Undersecretary of Defense for Research and Engineering

(B28) ../rfc/rfc791.pdf

Bu dosya (rfc793.pdf), belgenin XML biçiminin TEXLive ve belgeler-xsl paketlerindeki araçlar kullanılarak PDF biçimine dönüştürülmesiyle elde edilmiştir.

17 Ocak 2007