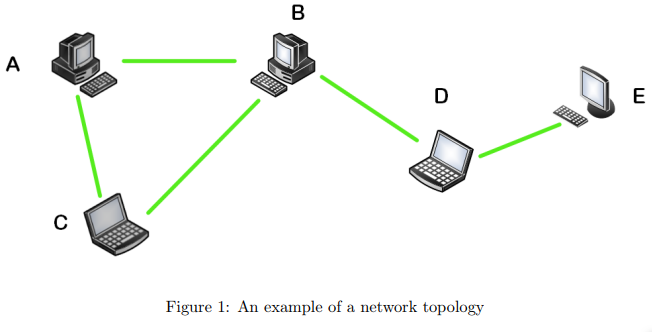
: ASSIGNMENT 2 :

1. **Hedefler**

Göreviniz ağdaki eşler arasındaki ağ iletişiminin basit bir sürümünü tasarlamaktır; Yani, Internet protokolü paketine benzeyen oldukça basitleştirilmiş bir bilgisayar ağ protokolü ailesi uygulamak. Ağ protokol yığını, her biri aşağıdaki Tabloda gösterildiği gibi farklı amaçlara sahip olacak şekilde dört katmandan oluşacaktır:

|  |  |
| --- | --- |
| KATMAN | PROTOKOL HiZMETLERi |
| Uygulama | Ağı kullanan uygulamalar arasındaki iletişim. |
| Ulaştırma | Uçtan uca veri teslimi. |
| Ağ | Paketleri tanımlar ve yönlendirme sağlar. |
| Veri Bağlantısı / Fiziksel | Fiziksel medyaya erişim için rutinler |

Ağın eşler arası bir modeli olacaktır. Aşağıdaki şekilde bir örnek ağ topolojisi gösterilmiştir.



Amaç, bir grup protokolle uyuşurken mesaj alışverişi yaparak arkadaşların birbirleriyle iletişim kurabilmeleri için akranları (onları istemciler olarak göreceğiz) mümkün kılmaktır.

* 1. **Temel Katman Hizmetleri**

• Bir istemci uygulaması, isteğe bağlı bir uzunlukta bir mesajı ağdaki başka bir istemciye göndermek için bir istekte bulunur. İleti iletişim ağındaki bağlantılar, fiziksel ortam üzerinde tek bir çerçeve içinde iletilebilen sınırlı bir mesaj boyutuna sahip olacağından, mesajın izin verilen boyuttaki mesaj parçalarına bölünmesinden Uygulama Katmanı sorumlu olacaktır. Bu noktada, gönderenin ve alıcının kimlikleri, iletişim kurulacak tarafları belirlemek için uygun olacaktır. Her bir mesaj parçası ayrı bir çerçevede gönderilecektir.

• Aktarım Katmanı, hangi uygulamanın iletişim kurmaya çalıştığına ve giden veya gelen bir soket olup olmadığına bağlı olarak giden bağlantı noktalarına doğru bağlantı noktası numaralarının eklenmesinden sorumludur.

• Ağ Katmanı, gönderenin ve alıcının IP adresleriyle ilgili bilgileri çerçeveye eklemelidir.

• Veri Bağlantısı / Fiziksel Katman, sonunda, çerçevelerin bir sonraki sekmeyi yapacakları bağlantının her iki ucunda bulunan fiziksel MAC adreslerini ekleyecektir.

* 1. **Ağ Hizmetlerinin Özeti**

Uygulamanız gereken ağ hizmetlerinin bir özeti şöyledir:

• Bir ağ içindeki istemciler arasında iletişim kuralı kümesini temel alarak iletişimi etkinleştirin.

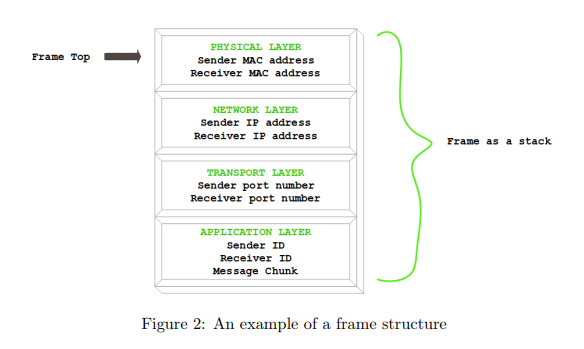
• Müşterilerin yönlendirme tablolarını kullanarak veri paketlerinin basit bir şekilde yönlendirilmesini sağlayın.

• Tüm ağ aktivitelerinin günlüğe kaydetmesini etkinleştir.

• Günlük raporlarını gerektiği gibi yazdırın.

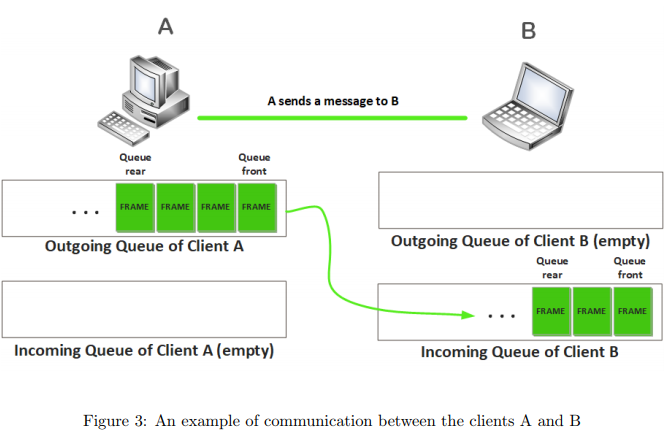
* 1. **Ağ Protokolleri**
     1. Yığın olarak Çerçeveler (İlk Giren İlk Çıkar)

Mesaj parçalarını taşıyan veri paketleri gönderilmeden önce, her bir ağ katmanı ile ilgili bilgi yığınları olarak uygulanması gereken Veri Bağlantısı / Fiziksel katman çerçeveleri içine kapsüllenmiş olmalıdır (örneğin, uygulama katmanı için gönderici ve alıcı kimlikleri içeren mesaj parçası) Ağ protokolü tarafından tanımlanan ağ katmanı, aktarım katmanı için port numaraları, fiziksel katman için fiziksel MAC adresleri, vs.) gönderici ve alıcının IP adresleri. Aşağıdaki şekilde bir çerçeve yapısı gösterilmiştir.



* + 1. **Giden ve Gelen Kuyruklar (İlk Giren İlk Çıkar)**

Her bir müşteri, giden ve gelen mesajlar için sırasıyla tamponlar (kuyruk olarak uygulanması gereken) gönderir ve alır (aşağıdaki resme bakınız). Tampon boyutunun sonsuz olduğunu varsayabilirsiniz; yani, bir sıraya yeterli yer olmadığı için hiçbir paketin bırakılmaması gerekir; daha ziyade, sıra, yerleştirilmesi gereken herhangi bir çerçeveyi barındırmak için gerektiği kadar genişleyecektir. Tüm kuyruklar tarafından kullanılan belleği gerektiğinde dinamik olarak ayırmanız ve (yeniden tahsis veya boşaltma) yapmanız gerekir.



Yukarıdaki şekil, A müşterisi ile A'nın mesajlaşma B olduğu bir istemci B arasındaki bir örnek iletişimi göstermektedir. Mesaj, her biri bir mesaj fragmanı taşıyan çerçevelerde kapsüllenir (mesaj boyutu izin verilen maksimum mesaj uzunluğunu aştığında, bir mesaj birden fazla kareye gönderilecektir). Çerçeve, gönderenin giden sırasından ayrıldığında, alıcının gelen sırasına eklenir. Çerçeveler FIFO yöntemini takip etmelidir (ilk mesaj, ilk çerçeveye gönderilecek ve alıcının tarafında ilk çerçevede alınacaktır).

* + 1. **Yönlendirme**

Bir müşteri gelen sıraya mesaj çerçeveleri aldığında, hedeflenen alıcının kim olduğunu kontrol etmelidir. Çerçeve, alıcı istemciye yönlendirilirse, son ileti parçası geldiğinde ileti yığınlarını paketinden çıkarmaya devam edebilir ve mesajı yeniden birleştirmek için dinleme soketinin bağlantı noktasına göre uygulama katmanına geçirebilir. . Ancak, iletinin başka biri için tasarlanması durumunda, istemcinin hedeflerine ulaşması gereken bir sonraki sekmeyi belirlemek için istemci, yönlendirme tablosunu kontrol etmelidir. Her müşterinin yönlendirme tabloları aşağıdaki biçime sahip olacaktır:

|  |  |
| --- | --- |
| İstenilen Hedef | Paketin iletilmesi gereken komşu |

Yönlendirme tabloları, ağdaki her istemci için bir tane olmak üzere birden çok satıra sahip olacaktır. Örneğin, giriş | A | B | Bir müşterinin A'ya ulaşması gerekirse, çerçeveyi B'ye göndermesi gerektiği anlamına gelir. Aşağıdaki şekilde bir yönlendirme örneği gösterilmektedir. Bu senaryoda, istemci A bir ileti istemcisi D göndermeye çalışıyor. İleti, izin verilen maksimum mesaj uzunluğundan daha uzun, bu yüzden dört kareye bölünmüş durumda. A için yönlendirme tablosu girdisini denetler. A, D'ye doğrudan bir bağa sahip olmadığından, yönlendirme tablosu, hangi istemcinin, sonunda D'ye erişebilecek bir yol üzerinde olduğunu belirtir ve bu tür bir istemci varsa (bu durumda, istemci B'dir). A, iletiyi iletmek için bu istemciye gönderir. Bir sonraki atlama ile ilgili bilgiler bir nedenle eksikse, mesaj bırakılır. B, D için öngörülen mesajı aldığında, mesaj çerçeveleri için bir sonraki hedefi belirlemek için B'nin yönlendirme tablosunu kontrol etmesi gerekir. Bu durumda, D, B'nin (B'nin yönlendirme tablosu girişinden bellidir) doğrudan bir komşudur, bu nedenle B mesajı D'ye iletir. Çerçeveler, son hedefe ulaşmadan önce ağ üzerinde birden fazla atlama yapmak zorunda kalabilir. Toplam atlama sayısı ile ilgili bilgilerin de günlüklere kaydedilmesi gerekir.

* + 1. **Kayıtlar**

Her ağ etkinliği günlüğe kaydedilmelidir. Böylelikle, her müşterinin gönderilen, alınan ve iletilen mesajlar hakkındaki bilgileri depolayan kendi kaydı olacaktır. Her bir günlük girişi aşağıdaki bilgileri saklamalıdır:

• Zaman damgası: etkinliğin tarihi ve saati,

• Çerçevelerde taşınan mesajlar,

• Bütün mesajı taşıyan toplam kare sayısı,

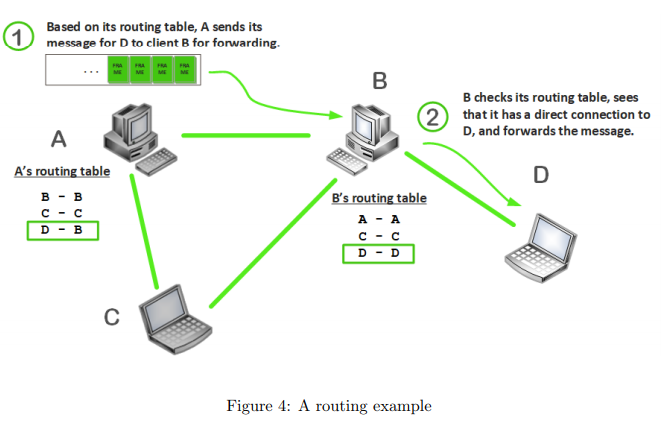
• Şebekenin şu ana kadar yaptığı toplam atlama sayısı,

• Gönderici kimliği,

• Alıcı Kimliği

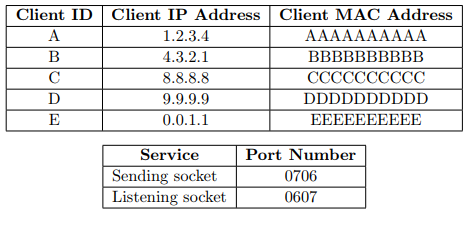
• Etkinlik türü (gönderilen, gönderilen veya iletilen mesajlar) ve

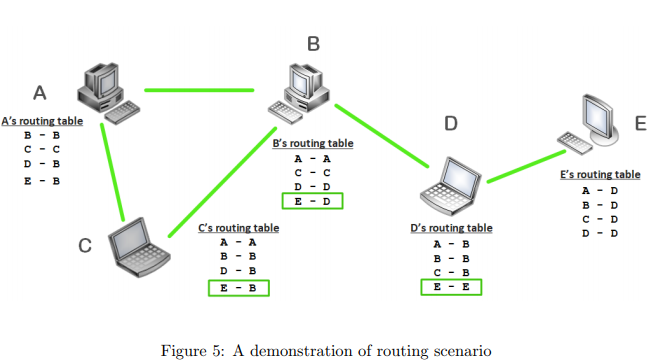
• Etkinlik başarı durumu (başarılı veya başarısız).



1. **Örnek Kullanım Örneği**

Şekil 1'de gösterilen bir ağımız olduğunu varsayalım ki, bu linkler tarafından desteklenen maksimum mesaj uzunluğu 20 karakterdir. Her müşterinin yönlendirme tabloları aşağıda gösterildiği gibi olacaktır. Ayrıca, istemci kimlikleri, adresler ve soket bağlantı noktası numaralarının aşağıdaki gibi olduğunu varsayalım.





İstemcinin C'nin E istemcisiyle iletişim kurmasını ve aşağıdaki içerikle birlikte mesaj göndermesini istiyoruz: “Çerçeveler için birkaç küçük atlama, ancak bu mesaj için dev bir sıçrama.” Mesajın uzunluğu 63 karakter ve maksimum olduğu için desteklenen kare mesaj uzunluğu 20, mesaj aşağıdaki mesaj parçaları ile dört farklı çerçeve üzerinden gönderilmesi gerekir (boşlukların da karakter olarak sayıldığını unutmayın):

1. “A few small hops for”

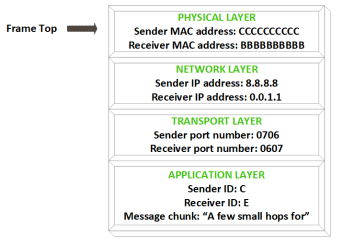
2. “ frames, but a giant”

3. “ leap for this messa”

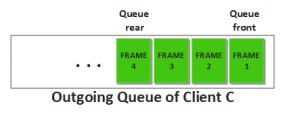
4. “ge.”

Bu iletişimin gerçekleşmesi için ağ içinde aşağıdaki adımların gerçekleştirilmesi gerekecektir:

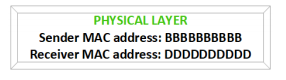
1. C, iletiyi parçalara ayırır ve gönderilecek dört çerçeve içinde kaplar ve çerçeveleri giden sıraya yerleştirir. Örneğin, ilk çerçeve aşağıdaki biçime sahip olacaktır:



Kalan üç çerçeve, taşıdıkları mesaj yığınındaki tek farkla benzer olacaktır. Bu işlemden sonra, C istemcisinin giden sırası aşağıda gösterildiği gibi görünecektir.



1. C, sonraki alıcıyı belirlemek için yönlendirme tablosunu kontrol eder. C’nin yönlendirme tablosundan, E'nin E’nin doğrudan komşusu olmadığı için, E için tasarlanan paketler için bir sonraki sekmenin istemci B olması gerektiği açıktır. Bu nedenle, C, çerçevelerin B-linkine bir sonraki atlamayı yapacağı gerçeğini yansıtacak şekilde, Fiziksel katman bilgisini (MAC adreslerini) uygun şekilde ayarlayarak, çerçeveleri B'ye gönderecektir.
2. B, gelen sıradaki çerçeveleri C'nin giden kuyruğundan çıkarıldıkları sırayla alır. Bu işlem sıra işlemleri kullanılarak yapılmalıdır; Diğer bir deyişle, gönderenin giden kuyruğundan bir çerçeve kaldırıldığında, tüm çerçeveler aktarılana kadar alıcının gelen sırasına eklenir.
3. B, istenen alıcıyı kontrol etmek için alıcı bilgilerini çerçeveler içerisinde kontrol edecektir. Mesajın, ağdaki başka bir istemci için, yani E için tasarlandığını görecektir. Böylece, B gelen kareleri gelen kuyruğundan çıkaracak ve iletilmek üzere giden kuyruğuna yerleştirecektir.
4. B, çerçevelerin bir sonraki atlayışını belirlemek için yönlendirme tablosunu kontrol edecek ve çerçevelerin D'ye iletilmesi gerektiğini görecektir. Bu noktada, çerçevelerin MAC adresleri, çerçevelerin gerçeğini yansıtacak şekilde yeniden ayarlanmalıdır. Bir sonraki bağlantıyı BD linki üzerinden yapacak. Her çerçevenin fiziksel bir tabakası olur: ------->>>



1. Bir önceki başvuru durumuna benzer şekilde, D, gelen sıradaki çerçeveleri B'nin giden kuyruğundan çıkarıldıkları sırayla alır.
2. D, istenen alıcıyı kontrol etmek için alıcı bilgilerini çerçeveler içerisinde inceleyecektir. Mesajın E için tasarlandığını görecek, böylece D gelen kareleri gelen sıradan çıkaracak ve iletilmek üzere giden kuyruğuna yerleştirecektir.
3. D, çerçevelerin bir sonraki atlayışını belirlemek için yönlendirme tablosunu kontrol edecek ve E'nin doğrudan komşusu olduğunu görecektir. Bu nedenle, D, çerçeveleri benzer bir şekilde E'ye iletecektir (tekrar MAC adreslerini gerektiği gibi ayarlayarak).
4. Son olarak, E, gelen karedeki kareleri aldığında, mesajın hedeflenen hedefe ulaştığını fark edecektir. E, iletinin paketini açacak ve uygulamasına iletecektir.

Her aktivitenin, gönderi, alım veya yönlendirme olup olmadığı, ilgili istemci tarafından yukarıda açıklanan biçimde kaydedilmelidir. Özetlemek gerekirse, çerçevelerin alınması durumunda, müşterilerin ilk önce istenilen alıcıyı kontrol etmesi gerekir. Eğer mesaj varış noktasına ulaştıysa, paketten çıkarılabilir, aksi halde iletilmelidir. Diğer taraftan, gönderim durumunda, çerçeveler için bir sonraki sekmeyi belirlemek üzere yönlendirme tablolarına başvurulmalıdır.

1. **Giriş Dosyaları ve Parametreleri**

Üç giriş dosyası (ikisi, ağın başlatılması için iki ve komutlar ile bir tane) ve sırasıyla iletilecek uygulamalar için desteklenen maksimum mesaj boyutunu, giden ve gelen port numaralarını belirten üç ek argüman olacaktır. İstemciler ve ağ topolojisi ilk iki giriş dosyası aracılığıyla başlatılacaktır.

* 1. **İstemci Bilgi Dosyası Biçimi**

***number\_of\_clients***

***Client\_ID<space>Client\_IP\_Address<space>Client\_MAC\_Address***

Örneğin, örnek senaryodaki beş istemciyle bir ağı başlatmak için giriş dosyası aşağıdaki içeriğe sahip olacaktır:

5

A 1.2.3.4 AAAAAAAAAA

B 4.3.2.1 BBBBBBBBBB

C 8.8.8.8 CCCCCCCCCC

D 9.9.9.9 DDDDDDDDDD

E 0.0.1.1 EEEEEEEEEE

* 1. **Yönlendirme Tabloları Dosya Biçimi**

***Client\_1\_routing\_table***

***-***

***Client\_2\_routing\_table***

***-***

***...***

***-***

***Client\_n\_routing\_table***

Her bir istemci yönlendirme tablosunun aşağıdaki biçimi vardır:

“Destination\_Client\_ID<space>Client\_ID\_of\_nearest\_neighbor”

Örneğin, örnek kullanım durum senaryosunda aşağıdaki yönlendirme bilgisi dosyası içerikleri bulunur:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 |
| B B  C C  D B  E B | A A  C C  D D  E D | A A  B B  D B  E B | A B  B B  C B  E E | A D  B D  C D  D D |

* 1. **Komut Dosyası Biçimi**

Çalışmalarınız üçüncü giriş dosyasında verilecek komutlarla test edilecektir. Komutlar, programınızın simüle etmesi gereken ağ etkinliğini belirleyecektir (örneğin, hangi istemcinin kiminle ve hangi iletiyle birlikte iletilmesi, ağ durumuyla ilgili hangi çıktının bekleneceği, vb.). Dosya formatı aşağıda verilmiştir.

***number\_of\_commands***

***COMMAND<space>command\_parameter\_1<space>command\_parameter\_2...***

Örnek bir komut dosyası ve komutların nasıl yürütüleceği ile ilgili talimatlar için, Ağ Durumu Denetimi Komutları bölümüne bakın.

* 1. **Program Yürütme**

Programınız yürütüldüğünde, tüm bu parametreler aşağıdaki sırada komut satırı argümanları olarak verilecektir:

***.\HUBBMNET clients.dat routing.dat commands.dat max\_msg\_size outgoing\_port incoming\_port***

Dosya adlarının değişebileceğini unutmayın, ancak sipariş her zaman aynı kalacaktır. Programınız bunu karşılayabilmelidir.

1. **Ağ Durumu Kontrol Komutları**

Yukarıda tartışılan örnek kullanım örneğinde bir ağ durumu kontrolü talep edersek, bunu gerçekleştirecek örnek bir komut giriş dosyası ve beklenen çıktı.

***commands.dat content:***

***---------------------------***

***7***

***MESSAGE C E #A few small hops for frames, but a giant leap for this message.#***

***SHOW\_FRAME\_INFO C out 3***

***SHOW\_Q\_INFO C out***

***SHOW\_Q\_INFO C in***

***SHOW\_FRAME\_INFO C in 5***

***SEND C***

***PRINT\_LOG D***

Komutların açıklaması aşağıda verilmiştir.

* 1. **Müşteri iletişim**

İki müşteri arasında bir mesaj alışverişini simüle etmek istediğimizi varsayalım. MESAJ komutu iletilecek mesajın hazırlanmasını tamamlayacak; yani parçalara bölünmesi. MESSAGE komutunun biçimi şöyledir:

***MESSAGE<space>sender\_ID<space>receiver\_ID<space>#message#***

Bu komut iletinin hazırlanmasını (iletide yer almaması gereken iki karma etiket arasında verilen), gerektiğinde parçalara ayırarak iletimi iletmek için tetikler ve çerçeveleri gönderenin giden kuyruğuna yerleştirir. Mesaj, GÖNDER komutu alınana kadar gönderilmemelidir. Örnek MESSAGE komutumuzun çıkışı, iletiyle ilgili bilgileri ve iletimi için hazırlanan tüm çerçeveleri aşağıda belirtilen biçimde görüntülemelidir.

***---------------------------------------------------------------------------------------***

***Command: MESSAGE C E #A few small hops for frames, but a giant leap for this message.#***

***---------------------------------------------------------------------------------------***

***Message to be sent: A few small hops for frames, but a giant leap for this message.***

***Frame #1***

***Sender MAC address: CCCCCCCCCC, Receiver MAC address: BBBBBBBBBB***

***Sender IP address: 8.8.8.8, Receiver IP address: 0.0.1.1***

***Sender port number: 0706, Receiver port number: 0607***

***Sender ID: C, Receiver ID: E***

***Message chunk carried: A few small hops for***

***--------***

***Frame #2***

***Sender MAC address: CCCCCCCCCC, Receiver MAC address: BBBBBBBBBB***

***Sender IP address: 8.8.8.8, Receiver IP address: 0.0.1.1***

***Sender port number: 0706, Receiver port number: 0607***

***Sender ID: C, Receiver ID: E***

***Message chunk carried: frames, but a giant***

***--------***

***Frame #3***

***Sender MAC address: CCCCCCCCCC, Receiver MAC address: BBBBBBBBBB***

***Sender IP address: 8.8.8.8, Receiver IP address: 0.0.1.1***

***Sender port number: 0706, Receiver port number: 0607***

***Sender ID: C, Receiver ID: E***

***Message chunk carried: leap for this messa***

***--------***

***Frame #4***

***Sender MAC address: CCCCCCCCCC, Receiver MAC address: BBBBBBBBBB***

***Sender IP address: 8.8.8.8, Receiver IP address: 0.0.1.1***

***Sender port number: 0706, Receiver port number: 0607***

***Sender ID: C, Receiver ID: E***

***Message chunk carried: ge.***

Veri Bağlantısı / Fiziksel katmandaki (MAC adresi) alıcı bilgisinin, alıcı IP adresi ve kimliği ile uyuşmadığını unutmayın. Bunun nedeni, bu mesajın hedeflenen alıcının E alıcısı olmasıdır. Ancak, müşteri E, C müşterisinin doğrudan bir komşusu değildir, bu nedenle C, bu mesajı en sonunda iletiyi doğrudan iletebilen doğrudan komşularından birine göndermelidir. son varış yeri. Bu bilgi C’nin yönlendirme tablosunda sağlanır. C, iletinin E. tarafına iletilebilecek bağlantının diğer tarafında kimin olduğunu görmek için yönlendirme tablosunu kontrol etmelidir. Örnek kullanım örneğimizde, sonraki atlama komşusu, istemci B'dir; Bu nedenle, Veri Bağlantısı / Fiziksel katman bilgisi buna göre ayarlanmalıdır. İletinin son hedefi istemci E olsa bile, iletinin C-B bağlantısı üzerinden ağda bir sonraki sekmeyi alması gerekir ve bu katman IP adreslerini veya istemcinin kimliklerini umursamaz. Sadece bu noktada çerçevelerin gidecekleri bağlantının her iki ucundaki istemcilerin MAC adreslerini önemser.

* 1. **Gelen / Giden Kuyruk Durumu**

İleti gönderme isteğini ileten iletişim uygulaması gibi Çerçeve # 3'ün içeriklerini C’nin Giden Kuyruk Sırasında incelemek istediğimizi varsayalım. Komut SHOW FRAME INFO bunu başaracaktır. Onun formatı:

***SHOW\_FRAME\_INFO<space>client\_ID<space>which\_queue<space>frame\_number***

Sırasıyla giden ve gelen kuyruğu belirten sıraya ya da dışa açılabilen parametre. Bu komut yerleştirildiğinde görüntülenmesi gereken bilgiler aşağıdaki gibidir:

***---------------------------------***

***Command: SHOW\_FRAME\_INFO C out 3***

***---------------------------------***

***Current Frame #3 on the outgoing queue of client C***

***Carried Message: " leap for this messa"***

***Layer 0 info: Sender ID: C, Receiver ID: E***

***Layer 1 info: Sender port number: 0706, Receiver port number: 0607***

***Layer 2 info: Sender IP address: 8.8.8.8, Receiver IP address: 0.0.1.1***

***Layer 3 info: Sender MAC address: CCCCCCCCCC, Receiver MAC address: BBBBBBBBBB***

***Number of hops so far: 0***

SHOW Q INFO komutu, genel olarak bir müşterinin kuyruğunu incelemek için kullanılır. Onun formatı:

***SHOW\_Q\_INFO<space>client\_ID<space>which\_queue***

Örneğimizde, C'nin giden sıra durumunu üçüncü komutla incelemek istiyoruz. Gerekli çıktı aşağıdaki gibi olmalıdır:

***---------------------------***

***Command: SHOW\_Q\_INFO C out***

***---------------------------***

***Client C Outgoing Queue Status***

***Current total number of frames: 4***

Benzer şekilde, dördüncü komutla, C’nin gelen sıranın durumu görüntülenir (bu noktada C'nin hiçbir şey almadığına dikkat edin):

***--------------------------***

***Command: SHOW\_Q\_INFO C in***

***--------------------------***

***Client C Incoming Queue Status***

***Current total number of frames: 0***

Beşinci komutla, C # 'nın gelen kuyruğunda Frame # 5'in içeriğini incelemeyi istedik. Çıkış şöyle olmalıdır:

***--------------------------------***

***Command: SHOW\_FRAME\_INFO C in 5***

***--------------------------------***

***No such frame.***

* 1. **Mesaj gönderme**

SEND komutu, gönderenin giden sırasına gönderilmekte olan mesajın gönderilmekte olan alıcısına gönderilmesine neden olacaktır. Onun formatı:

GÖNDER <boşluk> CLIENT\_ID

Bu komut örnek kullanım durumu senaryosunda açıklandığı gibi, son hedeflerine ulaşana kadar göndericinin giden kuyruğundan çerçevelerin iletimini ve iletimini tetikleyecektir. Aşağıdaki ağ izleme çıktısı üretilmelidir:

***----------------***

***Command: SEND C***

***----------------***

***A message received by client B, but intended for client E. Forwarding...***

***Frame #1 MAC address change: New sender MAC BBBBBBBBBB, new receiver MAC DDDDDDDDDD***

***Frame #2 MAC address change: New sender MAC BBBBBBBBBB, new receiver MAC DDDDDDDDDD***

***Frame #3 MAC address change: New sender MAC BBBBBBBBBB, new receiver MAC DDDDDDDDDD***

***Frame #4 MAC address change: New sender MAC BBBBBBBBBB, new receiver MAC DDDDDDDDDD***

***A message received by client D, but intended for client E. Forwarding...***

***Frame #1 MAC address change: New sender MAC DDDDDDDDDD, new receiver MAC EEEEEEEEEE***

***Frame #2 MAC address change: New sender MAC DDDDDDDDDD, new receiver MAC EEEEEEEEEE***

***Frame #3 MAC address change: New sender MAC DDDDDDDDDD, new receiver MAC EEEEEEEEEE***

***Frame #4 MAC address change: New sender MAC DDDDDDDDDD, new receiver MAC EEEEEEEEEE***

***A message received by client E from client C after a total of 3 hops.***

***Message: A few small hops for frames, but a giant leap for this message.***

Bu adımdaki en önemli şey, her linkin üzerinden atladıkça çerçeveler içinde uygun MAC adreslerinin ayarlanmasıdır. Örnek kullanım örneğindeki çerçeveler, C müşterisinden E yolculuğuna yolculuğuna başladıklarında, ilk sıçramaları C-B bağlantısı üzerindeydi. Bu nedenle, sırasıyla CCCCCCCCCC ve BBBBBBBBBB'nin bulunduğu göndericinin ve alıcının MAC adresleri. B kareleri aldığında, aslında istemcinin E için tasarlandığını ve bunları istemci D'ye ilettiğini görür, ilk önce kareleri paketlerinden çıkarmalı, Veri Bağlantısını / Fiziksel Katmanları her kareden çıkarmalı, MAC adresi bilgilerini değiştirecek şekilde değiştirmelidir. Her bir çerçevenin göndericisinin ve alıcının MAC adreslerini sırasıyla BBBBBBBBBB ve DDDDDDDDDDD olarak ayarlayarak ve son olarak da her bir çerçeveye geri getirilen Veri Bağlantısı / Fiziksel Katmanı tekrar bastırarak BD bağlantısı üzerinden bir sonraki sıçrama. İletiyi hedeflenen varış yerine iletmek her zaman mümkün olmayabilir. Örnek örneğimizde kullanım durumunun D'nin yönlendirme tablosunun aşağıdaki gibi olduğunu varsayalım:

***A B***

***B B***

***C B***

***E -***

Yani, müşteri E için tasarlanan mesaj, müşteri D'ye ulaştığında, D, bir sonraki atlamanın ne olması gerektiğini bulamazdı. Bu durumda, kullanıcının ulaşılamayacağını belirten bir hata mesajı da görüntülenmelidir. Bu durumda SEND komutunun tam çıktısı aşağıdaki gibi olmalıdır:

***----------------***

***Command: SEND C***

***----------------***

***A message received by client B, but intended for client E. Forwarding...***

***Frame #1 MAC address change: New sender MAC BBBBBBBBBB, new receiver MAC DDDDDDDDDD***

***Frame #2 MAC address change: New sender MAC BBBBBBBBBB, new receiver MAC DDDDDDDDDD***

***Frame #3 MAC address change: New sender MAC BBBBBBBBBB, new receiver MAC DDDDDDDDDD***

***Frame #4 MAC address change: New sender MAC BBBBBBBBBB, new receiver MAC DDDDDDDDDD***

***A message received by client D, but intended for client E. Forwarding...***

***Error: Unreachable destination. Packets are dropped after 2 hops!***

Gönderici istemciden son hedefe (amaçlanan alıcı) ulaşana kadar mesaj gönderme işlemi, yinelemeli olarak gerçekleştirilmelidir. Mesaj çerçeveleri, hedeflenen istemciye ulaşana kadar ağ üzerinden geçmeli veya bazı öngörülemeyen durumlar (örneğin ulaşılamaz kullanıcı) nedeniyle düşmelidir.

* 1. **Müşteri Günlükleri**

Son komut PRINT LOG ile birlikte, İstemci D'nin günlüğünü

C ve E arasındaki mesajın iletilmesinin başarılı bir şekilde tamamlanması. Bu komutun formatı:

***PRINT\_LOG<space>client\_ID***

***Bu noktaya kadar, D bir mesaj aldı ve iletti, böylece çıktı şöyle olmalıdır:***

***---------------------***

***Command: PRINT\_LOG D***

***---------------------***

***Client D Logs:***

***--------------***

***Log Entry #1:***

***Timestamp: 2018-9-2 15:59:16***

***Message: A few small hops for frames, but a giant leap for this message.***

***Number of frames: 4***

***Number of hops: 2***

***Sender ID: C***

***Receiver ID: E***

***Activity: Message Received***

***Success: Yes***

***--------------***

***Log Entry #2:***

***Timestamp: 2018-9-2 15:59:17***

***Message: A few small hops for frames, but a giant leap for this message.***

***Number of frames: 4***

***Number of hops: 2***

***Sender ID: C***

***Receiver ID: E***

***Activity: Message Forwarded***

***Success: Yes***

Yönlendirme bilgilerinin eksikliğinden dolayı bir sonraki sıçrama belirlenemediğinden çerçevelerin düşürülmesi durumunda, D müşterisinin ikinci kayıt girişi:

***Log Entry #2:***

***Timestamp: 2018-9-7 11:57:22***

***Message: A few small hops for frames, but a giant leap for this message.***

***Number of frames: 4***

***Number of hops: 2***

***Sender ID: C***

***Receiver ID: E***

***Activity: Message Forwarded***

***Success: No***

İstemcinin henüz herhangi bir kayıt girişi olmaması durumunda, hiçbir şey basılmamalıdır.

* 1. **Geçersiz Komutlar**

Geçersiz bir komut sağlandığında, programınız aşağıdaki mesajı vermelidir:

***---------------------***

***Command: ORDER pizza***

***---------------------***

***Invalid command.***

1. **Önemli notlar**

Yığın ve sıra yapılarının uygulanması: Çerçeveler yığınlar halinde uygulanmalıdır (katmanlara bilgi eklemek ve çerçevelerden kaldırmak için pop ve itme yöntemleri kullanılmalıdır). Benzer şekilde, tüm kuyruklar sıra veri yapıları olarak uygulanmalıdır (sıralara ve kuyruklardan kareler eklemek ve kaldırmak için ekleme ve çıkarma yöntemleri kullanılmalıdır). Bunu sağlamak için kodunuz iyice kontrol edilecektir. Bu uygulamalardan yoksun olan işler, çıktıdan bağımsız olarak 0 puan ile derecelendirilecektir (not verme politikasını kontrol ediniz). Dinamik bellek tahsisi kullanımı: Gönderi kapasitelerinin yanı sıra, istemci günlüklerinin yanı sıra, dinamik olarak tahsis edilmesi ve yönetilmesi gerekecek. Bellek kaynaklarını boşa harcamamak için kodunuzda dinamik bellek ayırmayı kullanmamak, notunuzu otomatik olarak 50 puan düşürecektir. Çalışmanız nasıl derlenecek: Kodunuz gcc sürüm 4.8.5 kullanılarak dev üzerinde derlenecektir. Bu nedenle, göndermeden önce kodunuzda kodunuzu derlemeniz ve sınamanız önerilir. Ayrıca, kaynak kodlarınızla bir makefile eklemelisiniz. Kodunuz derlenecek ve dev üzerindeki aşağıdaki komutlar kullanılarak çalıştırılacaktır:

***$ make***

***$.\HUBBMNET clients.dat routing.dat commands.dat max\_msg\_size outgoing\_port incoming\_port***