**HTTP**

HTTP (hyper text transfer protocol), uygulama katmanında çalışan bir internet protokolüdür. İstemci-sunucu modeline göre işleyen bu protokol, web üzerinden bilginin sunucudan kullanıcıya nasıl ulaşacağını ve hangi şekilde yönlendirileceğini belirlemektedir. İstemci herhangi bir istekte bulunduğunda sunucuya bir http mesajı gönderilir ve sunucu bu mesaja karşılık bir http yanıt mesajı gönderir.

Kullanıcı, sunucuya bir http mesajı yolladığında istemci ve sunucu arasında bir bağlantı kurulur. Http mesajlarının iletimde kurulan bu bağlantılar ise ulaşım katmanında çalışan protokoller sayesinde kontrol edilir. Yani http bu bağlantılar üzerinde bir yetkiye sahip değildir. Http’nin bu durumda en sık kullandığı protokol ise TCP’dir. Aslında temel olarak http altta çalışan bir iletişim protokolüne ihtiyaç duymaz. Ancak bu protokolleri hem güvenilir olmak için hem de olabildiğince az veri kaybı yaşamak için kullanır.

Kurulan bağlantıların önemli bir özelliği ise kalıcı olmamalarıdır. İstemcinin gönderdiği http mesajı sunucuya vardığı anda kurulan bağlantı koparılır ve yanıt gönderilirken gereken bağlantı tekrar kurulur.

İstemci ve sunucunun gönderdiği http mesajları genel olarak benzeseler de aralarında bazı farklılıklar vardır. İlk olarak Http istek mesajı içerisinde kullanılmak istenen metot, http protokol versiyonu ve header bulunur.



http metodu, istemcinin sunucuda gerçekleşmesini istediği işlemi belirtir. http nin çeşitli versiyonlarında birden fazla metot bulunur. Bunlar arasından en çok kullanılanlardan birisi ‘GET’ dir. Bu metot, sunucudan istenilen verileri getirmekle görevlidir. Örneğin herhangi bir web sitesine girildiğinde o sitenin html dosyası ‘GET’ metodu ile alınır. ‘HEAD’, ‘POST’, ‘DELETE’, ‘PUT’ ise diğer metotlardan bir kaçıdır.

İstek mesajının header bölümünde ise ek bilgiler bulunmaktadır. Bu ek bilgiler, istenilen dil veya host bilgisi gibi birçok şey barındırabilir. Bu kısım http yanıt mesajında da bulunmaktadır.



Http yanıt mesajında http versiyonu (istek mesajında olduğu gibi), durum kodu, durum mesajı ve header bulunur. Durum kodu ve durum mesajları, sunucunun istemciye verdiği cevap kodları ve mesajlarıdır. Bunlar bazen hata kodu bazen isteğinin başarılı bir şekilde tamamlandığını belirten kodlar olabilir. En çok karşılaşılan hata kodlarından bazıları şunlardır:

* 404 Not Found
* 403 Forbidden
* 500 Internal server error
* 502 Bad Gateway

Eğer sunucu isteği başarı ile tamamlarsa bunu belirten durum kodu ile birlikte istenen işlem de gerçekleştirilmiş olur.

**HTTP SALDIRILARI VE ÖNLEMLERİ**

Her protokole olduğu gibi http protokolüne de birden fazla saldırı yapılabilmektedir. Bu saldırıların önemli bir kısmı ise TCP üzerinden yapılmaktadır. Bu nedenle http servisini koruma üzerinde çalışılacaksa gelebilecek saldırıların sadece http protokolü üzerinden gelmeyecek olduğunu unutmamak gerekir.

TCP üzerinden yapılan saldırılardan sonra en çok karşılaşılan saldırı ise HTTP flood saldırısıdır. Bu saldırı DDoS (distributed denial of service) tipinde bir saldırıdır ve sunucuya oldukça fazla istek göndererek bu sunucunun geçerli bir istek kabul etmesini engeller. Bu saldırı türünün iki çeşidi vardır. İlki ‘GET’ flood, ikincisi ‘POST’ flood.

GET flood saldırısında saldırgan, birden fazla makineden (daha etkili olması için bot da kullanılır) belirli bir servera GET metoduyla çeşitli isteklerde bulunurlar. Bu istekler web sayfasını açmak veya herhangi bir dosya istemek olabilir. Bunun sonucunda server bu istek yoğunluğuna cevap veremez ve trafiğe kapanmış olur.

POST flood da saldırının temel prensibi aynıdır. POST metodu, kendisine gönderilen verileri saklayabilir ve bu verileri genellikle bir databasede toplar. Bu saldırıda bu durumdan faydalanır. GET floodda olduğu gibi servera birden fazla makineden POST metodunu kullanarak bilgi gönderilir ve bu bilgilerin depolandığı database’in kapasitesi doldurulmaya çalışılır. Saldırı başarılı olursa da server yine trafiğe kapanır.

Bu saldırıları engellemek karmaşık olabilmektedir. Bunun nedeni, sunucuya gelen isteklerin hangisinin saldırgan hangisinin gerçek istek olduğunu ayırt etmenin oldukça zor olmasıdır. Bu saldırıyı önleyebilmek için birkaç tane yöntem kullanılabilir.

Bunlardan ilki herhangi bir istek yapılmadan önce istemcinin bot olup olmadığını anlamak için test yapmaktır. Bunun en yaygın örneği web sitelerine kaydolmadan önce kullanılan Captcha Test’tir. Diğeri ise IP tekrarını kontrol edebilmek için kullanılabilecek bir web application firewall kullanmaktır. Bu sayede aynı IP adresinden gelen istekler tespit edilip saldırılar önlenebilir.

HTTP saldırılarından bir diğeri ise low and slow saldırısıdır. Bu saldırı da HTTP flood saldırısı gibi DDoS türünde bir saldırıdır ancak flood a göre daha yavaş ve tespit edilmesi daha zor bir saldırıdır.

Saldırının temel prensibi thread tabanlı web sunucularının threadlerini çeşitli isteklerle tıkamak ve gerçek isteklerin alınmasını engellemektir. Bu işlem birkaç farklı yolla yapılabilir. Bunlardan bir tanesinde, HTTP headerlarının bir kısmını sunucuya gönderip bağlantı oluşturduktan sonra headerın geri kalan kısmını yavaş bir şekilde (time out olacak kadar yavaş değil) sunucuya iletilir. Bunun sonucunda sunucu tıkanır ve gerçek bir istek alamaz.

Bu saldırının avantajlarından biri diğer saldırılar gibi kaba kuvvet kullanmamasıdır. Bunun yerine tek bir makineden herhangi bir isteğin yavaş bir şekilde iletilmesi yeterlidir. Bu sayede bot kullanımına da gerek kalmaz.

Bu saldırıda sunucu isteklerle dolmak yerine sadece birkaçıyla tıkandığı için tespit edilmesi çok zordur. Tespit etmek için ise kullanılabilecek en iyi yöntem sunucu loglarının sürekli ve dikkatle incelenmesidir.

Saldırıyı tespit ettikten sonra engellemek de kolay değildir. Bunun için kullanabilecek yöntemlerden biri sunucunun kapasitesini arttırmaktır. Ancak bu yöntem de kalıcı değildir. Çünkü saldırgan bu durumu fark ettikten sonra saldırısının da ölçeğini arttırıp istediği sonuca ulaşabilir. Bir diğer yöntem ise reverse-proxy tabanlı koruma kullanmaktır. Bu sayede gelecek olan saldırı sunucuya varmadan engellenir.