Web Teknolojileri

Genel olarak web uygulamaları üç temel bileşenden oluşmaktadır.

İstemci : Büyük Oranda Web Tarayıcı

Sunucu : İstemciden gelen istekleri karşılayan bir sunucu.

Veritabanı : Verilerin alınıp kaydedildiği ve çağrıldığı veritabanı.

İstemci Tarafı

İstemci taraflı teknolojiler internet kullanıcıları tarafından günlük kullanılan teknolojilerdir. Firefox, Chrome, Opera, Safari gibi tarayıcılar ile birlikte kullanılan HTML, JavaScript, CSS bunlara örnek olabilir.

Sunucu

İstemcilere bilgi sunan ve istemciden gelen verilerin işlenmesini sağlayan teknolojilerdir.

Çok fazla ve farklı teknolojiler burada bulunur. Bundan doalyı zafiyet bulunma olasılığı çok yüksektir.

Başlıca sunucu taraflı teknolojiler :

Apache, Nginx, IIS gibi web sunucuları

Tomcat, JBoss, Oracle Application Server gibi uygulama sunucuları

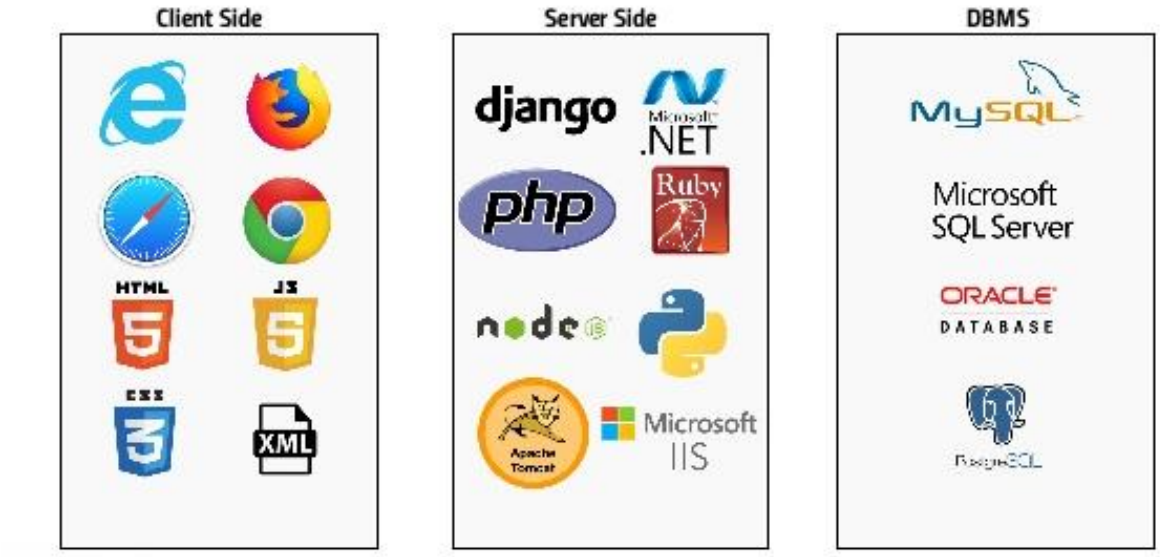
C#, PHP, Java, Python, Ruby, ASP başlıca kullanılan programlama dilleridir.

VeriTabanı

Veritabanları uygulama sunucularıyla aynı sunucu üzerinde bulunacağı gibi güvenlik sebebiyle farklı sunucular üzerinde de bulunabilir.

Başlıca database teknolojileri :

MySQL, Microsoft SQL Server, Oracle, PostgreSQL, NoSql,MongoDB,CouchDB



HTTP TEMELLERİ (Hyper-Text Transfer Protocol)

1990 Yılından beri internet üzerinde küresel bilgi paylaşımı için kullanılmaktadır.

HTTP, World Wide Web için veri iletişiminin temelidir; burada köprü metni belgeleri, örneğin bir fare tıklamasıyla veya bir web tarayıcısında ekrana dokunarak kullanıcının kolayca erişebileceği diğer kaynaklara köprüler içerir.

HTTP aynı zamanda SMTP,NNTP,FTP,Gopher ve WAIS iletişim kurallarını destekleyen internet sistemleri ile kullanıcı istemcileri ve vekil sunucular(Proxy) arasında iletişim için özelleştirilmiş bir iletişim kuralı olarak ta kullanılır.

İnternet ağında sunucular ve kullanıcılar arasında nasıl bir veri alışverişi olacağı hakkında kurallar vardır. Bu düzeni sağlayan protokol de HTTP'dir. Peki "Bu protokol günlük hayatta ne işimize yarar?" diye sorulacak olursa da aslında internette gezinmemizi sağlayan, internet sitelerini anında önümüze getiren bağlantı bu protokol sayesinde sağlanıyor.

Sistemin yürüdüğünü kısaca açıklayalım; girmek istediğiniz sitenin adresini tarayıcı çubuğunuza yazarsınız, bunu yazdıktan sonra HTTP yardımı ile siteye bir bağlantı isteği gider. Bu bağlantı isteğini kabul eden sitenin sunucusu ile bağlantı kurulur ve internet sitesine girmiş olursunuz. Eğer sunucudan bir cevap gelmezse karşınızda bir çeşit hata uyarısı alırsınız.

http protokolü genel olarak web tarayıcılarının kullandığı bir protokoldür.

Kullanıcı ve web uygulaması arasındaki trafik genellikle bu protokol üzerinden gerçekleşir.

Default olarak TCP 80 portunda çalışır.

Plaintext'dir.(içerisinde yalnızca yalın metinleri içeren metin dosyası veya dokümanlardır. Zengin metin belgelerinin aksine plaintext yazılarda kalın, italik, font değişikliği, font büyüklüğü gibi içeriği zenginleştirmeye yönelik değişiklikler gerçekleştirilemez. Bu nedenle plaintext dosyaları olduğu gibi yalın metinlerden ibarettir.)

İstek ve cevap yani , Request/Response şeklinde çalışır.

Durum kontrolü yapmadığı için Cookie ve Session mekanizmaları kullanılır.

Cevaplarında durum kodları bulunur.

Header ve Body olarak iki ana bölümden oluşur.

HTTP / 0.9 olarak bilinen ilk sürüm internet üzerinden ham(raw) verinin taşınması amaçlı, basit bir iletişim kuralıydı.

HTTP / 1.0 sürümü ile taşınan verinin meta-bilgilerini(meta-data) ve istek/cevap semantiğini düzenleyicilerini içeren ve MIME(Multipurpose Internet Mail Extensions; E-posta uygulamaları aracılığıyla gönderilecek olan iletiye çeşitli türdeki içeriği eklemek için kullanılan bir İnternet standartıdır.) ilgileri taşıyan mesajların taşınabilmesi gibi yenilikler eklenmiştir.

Apache ya da IIS gibi bir sunucunun üzerinde hizmet veren web sayfası HTTP protokolü sayesinde görüntülenir. HTTP bağlantısı yapılabilmesi için istemci HTTP REQUEST gönderir. Sunucu ise bu isteğe cevap olarak istemciye HTTP RESPONSE gönderir.

Örnek :

GET/HTTP 1.1

Host : facebook.com

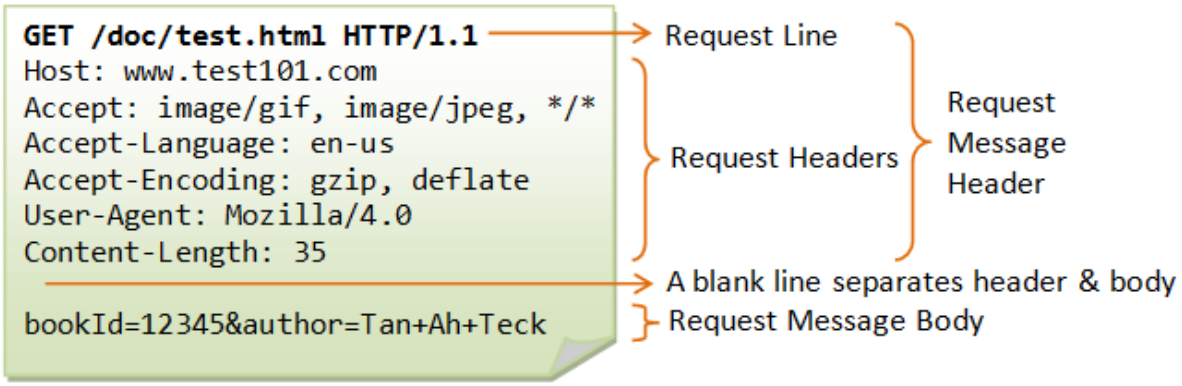
User-Agent : Mozilla/5.0 (Xll : Linux x86_64; rv:43.0) Gecko/20100101 Firefox/43.0 Iceweasel/43.0.4

Accept : text/html, application/xhtml+xml, application/xml;q=0.9,/*;q=0.8*

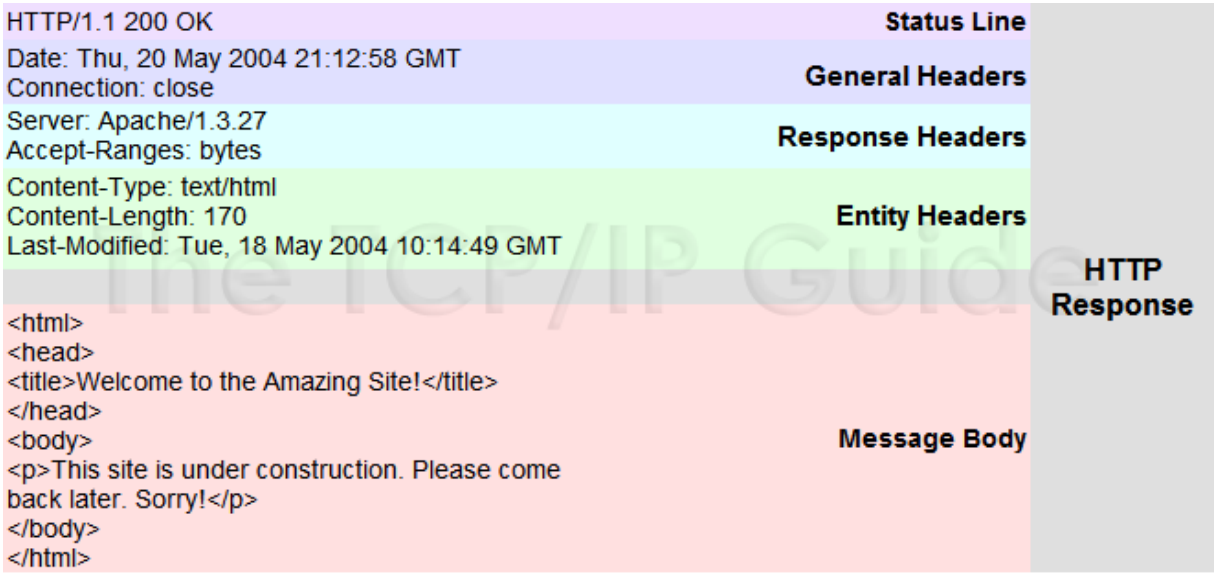
Accept-Language : en-US, en;q=0.5

Accept-Encoding : qzip, deflate

Connection : close



Response örneği :



200 OK – Bağlantı tamam mesajını verir.

300'lü mesajlar yönlendirme mesajları, 400 lü mesajlar istemci taraflı hata mesajları, 500 lü mesajlar ise sunucu taraflı hata mesajlarını gösterir.

HTTP METHOD

GET, POST, HEAD, PUT, DELETE, CONNECT, OPTIONS gibi birçok HTTP metodu vardır. Hepsinin amacı birbirinden farklıdır. Yapacağınız HTTP isteğinin türüne göre bu metotlardan birini kullanmalıyız

GET METODU

Sunucudan çok basit, ek bilgilere ihtiyaç duymayan, ihtiyaç duyuyorsa bile çok uzun olmayan, genelde statik bazen de dinamik sayfaları çağırmak için kullanılır.

POST METODU

Sunucu tarafında dinamik olarak oluşturulup, istemciye iletilecek HTML belgelerinde, dinamik veriler istemciden alınacağı zaman POST metodu kullanılır

Hem GET metodunda Hem de POST metodunda istemci sunucuya veri gönderebiliyor, ancak GET metodunda gönderilen verinin bir limiti vardır. POST metodunda ise limit yoktur.

HTTP REQUEST HEADERS

Client sunucuya istek yaptığında bazen kendi ile birlikte bazı bilgileri götürmesi gerekebilir. Bu bilgilere örnek olarak istemcinin kim olduğunu , nereden geldiğini, nereye gitmek istediğini verebiliriz.

Sürekli taşınacak bu bilgiler Request Header'lar ile kullanılır.

Host, Referer, Cookie, Authorization gibi başlıklar vardır.

Her header'ın ayrı bir görevi vardır.

HOST : Client'ın nereye gitmek istediğini belirten başlıktır.

Referer : Client'ın nereden geldiğini belirttiği başlıktır.

Cookie : Çerezlerin taşındığı başlıktır.

Authorization : Server tarafında oluşturulacak HTML belgeleri bir yetkilendirmeye göre otomatik olarak oluşturuluyor ise Client Server'a kim olduğunu bu başlık üzerinden belirtir.

detaylar : https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers

HTTP RESPONSE HEADERS

Sunucu istemciye hazırladığı yanıtı gönderirken bazen kendi ile birlikte bazı bilgileri götürmesi gerekebilir. Bu bilgilere örnek olarak istemcinin sunucunun hangi işletim sistemini kullandığını, sunucunun saatinin kaç olduğunu, sunucuda hangi yazılımların kurulu olduğu diyebiliriz.

Server,Date,Content-type,Location gibi başlıklar en çok kullanılanlardandır.

Server : Sunucunun hangi web sunucu yazılımını kullandığını istemciye belirtir.

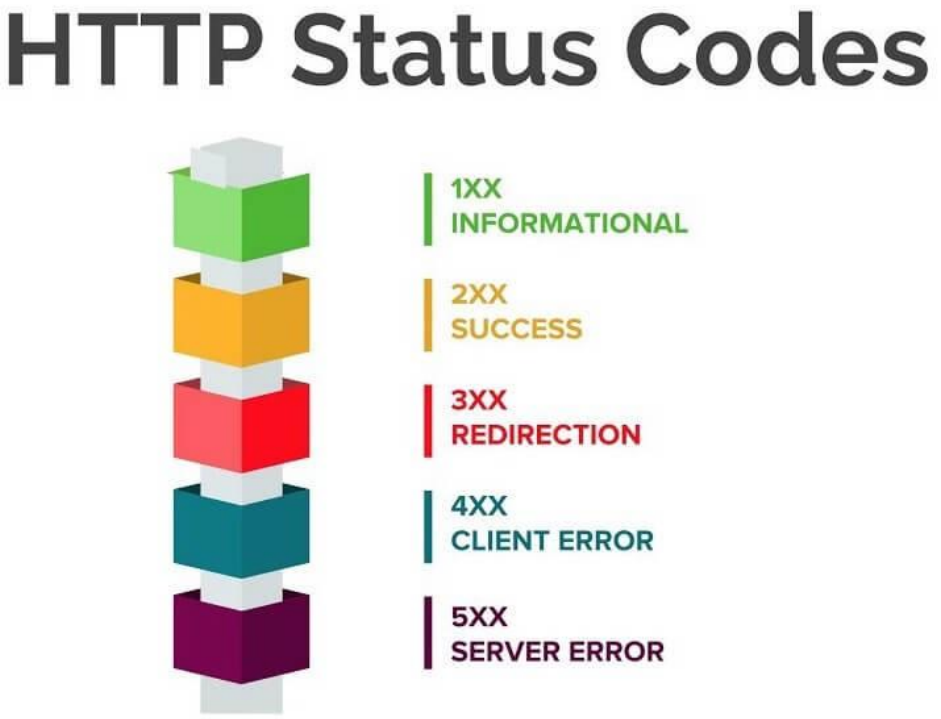
Date : Sunucunun tarih ve saat bilgilerini istemciye verir.

Content-type : Sunucunun istemciye ilettiği belgenin tipini belirtir.

Location : Sunucudan dönen yanıt her zaman bir sayfada olmayabilir. Eğer başka bir sayfaya yönlendirme varsa bu başlık ile belirtilir.

RESPONSE STATUS CODES

HTTP yanıt mesajlarını ilk satırda HTTP durum kodu vardır. Durum kodları 5 gruba ayrılır



HTTPS

HTTP Protokolü normalde veri transferlerinde şifrelenme kullanmadığı için araya giren 3.kişiler trafiği dinleyebiliyor ve bilgilerimizi çalabiliyordu. HTTPS(Security) protokolüyle defaultu TCP 443 protokolü üzerinden çalışır ve SSL TLS gibi protokolleri kullanarak trafiği **şifreli olarak sağlar.**

HTTPS ile XSS, SQL INJECTION gibi ataklar ÖNLENEMEZ. Haberleşmede gizlilik sağlamak için kullanılan bir protokoldür.

PROXY (VEKİL) SUNUCU

İnternet'e erişim sırasında kullanılan bir ara sunucudur. Bu durumda, örneğin bir ağ sayfasına erişim sırasında doğrudan bağlantı yerine:

Tarayıcı vekil sunucuya bağlanır ve hangi sayfayı istediğini söyler

Vekil sunucu gerekiyorsa o sayfaya bağlanır ve içeriği alır

Vekil sunucu tarayıcıya içeriği gönderir

Bilgisayar ağlarında, bir vekil sunucu diğer sunuculardan kaynakları isteyen istemcilerin talepleri için **bir aracı olarak davranan sunucudur.**

İstemci makine sunucu ile sunucu makine arasına girip, istemciden gelen isteği sunucuya iletip, sunucudan gelen cevabı ise istemciye ileten bir geçiş yolu (gateway) sistemidir.

Örneğin bir web sitesine giriyorsunuz ancak isteğin sizden gittiğini göstermek istemiyorsunuz, bir şekilde gizlilik sağlamak için Proxy server devreye giriyor. O site isteği alıp gerekli işlemleri yaptıktan sonra Proxy sunucusuna gerekli cevabı veriyor ve Proxy sunucuda cevabı alıyor. Sonrasında Proxy

artık sunucu gibi davranıyor ve siteden aldığı yanıtı size cevap olarak gönderiyor. Siz cevaba ulaşıyorsunuz ancak site sizden geldiğini görmüyor.

Bu teknoloji, birçok avantaj sağlar:

Fazladan hız: vekil sunucu, çok ziyaret edilen sayfaları önbelleğine alabilir. Bu durumda, o sayfa ziyaret edilmek istendiğinde dünyanın öbür ucundaki bir sunucuya bağlanmak yerine önbellekteki bilgi okunur.

Fazladan kontrol: vekil sunucu, istenen sayfalara erişim verip istenmeyenlere erişim vermeyebilir. Kimin hangi sayfaya girdiğini bellekte tutabilir. Gerekiyorsa, içeriği değiştirerek (örneğin küfürleri silerek) verebilir.

Fazladan güvenlik: vekil sunucu, virüslü dosyaları otomatik olarak temizleyebilir. Ayrıca, ağda hiç kimsenin İnternet'e doğrudan erişimi olmadığı için bir virüsü veya zararlı bir programı yayma ihtimalini de azaltır.

Mozilla Firefox 2.0 altında Vekil sunucu tanımlanması

Fazladan gizlilik: Özellikle Çinliler, Google ve Vikipedi gibi Çin hükümetince yasaklanan sitelere bağlanmak için bu yönteme başvururlar.

Asgari erişim: Kullanıcılar, özellikle hükûmet tarafından yasaklanan (teknik tabiriyle "erişimi engellenen") İnternet sitelerine bağlanmak için bu yönteme başvururlar.

Genelde İnternet servis sağlayıcılar, şirketler ve büyük ağlar (kampüs ağları gibi) tarafından kullanılır.

TRANSPARENT(SAYDAM) PROXY

Bu proxy'de tamamen gizlenme olmaz. İstekte bulunulan hedef sunucu gerçek istemcinin siz olduğunuzu bilir ancak yanıtı Proxy üzerinden gönderir.

"X-forwarded-for" adında bir HTTP header'ı ile sizin gerçek IP adresinizi de iletir.

Anonymous(Anonim)

Anonymous proxy server'lar web server olarak da adlandırılır. Internet ve LAN'larda çoğunlukla kullanılan proxy server'dır. Çoğunlukla web gezintilerinde iç ağdaki kullanıcıların IP adreslerini gizleyerek tek bir 1P adresinden erişim sağlar. Kullanıcı proxy server arkasında kalır. Ancak, erişilen internet sitesine erişim bilgileri olarak proxy server ve kullanıcının bilgileri gönderilir. İlk bakışta sadece proxy server bilgileri görünse de detaylı inceleme ile kullanıcının bilgileri de görülebilir.

High Anonymity Proxy

Elite proxy server olarak da adlandırılır. Bu tip proxy server'da bir siteye erişim sağlayan kullanıcının 1P adresi dışındaki bilgileri (kullandığı tarayıcı, işletim sistemi gibi) erişilen siteye gönderilir. Kullanım sayısı çok azdır. Bu tip server'lar kötü amaçlar ile kullanılmaya başlandığı için kullanımında bazı kısıtlamalar getirilmiş, bazı ülkelerde kullanımı özel izine bağlanmış ve kullanan kişilerin bilgilerinin 5 yıl saklanma zorunluluğu getirilmiştir.

SAME ORIGIN POLICY(Aynı Kök Politikası)

1994 Yılında HTTP protokolüne eklenen Cookie özelliği sayesinde web sayfaları durağanlığını üzerinden atmış, istekte bulunan kullanıcıları birbirinden ayırt edebilecek bir yeteneğe sahip olmuştu.

Webin kullanımı arttıkça daha fazla etkileşim ve çeşide ihtiyaç duyulmuştur. Cookie'nin ardından iyi yeni özellik Netscape tarafından tarayıcılara eklendi. Bunlardan biri istemci tarafından yüklenen kodların browserda yorumlanmasını sağlayan JavaScript dili , diğeri ise hiyerarşik bir metin dosyasından ibaret olan HTML dökümanlarının bu scripting dili ile etkileşimini sağlayan bir API olan DOM yani Document Object Model,Doküman Nesne Modeli.

DOM ile birlikte JScript kullanılarak doküman URL'inden barındırdığı Cookielere kadar dökümanın yüklenme olayından kapatılmasına kadar, HTML dökümanını ilgilendiren tüm özelliklere erişmek ve değiştirmek mümkün oldu.

HTTP nin zenginliği farklı kaynaklardan yüklenen içeriklerin istemci tarafında derlenmesinden kaynaklanır,dolayısıyla bir çok sorun ortaya çıktı.

Bu sorunla birlikte her bir kaynağın kendine has Cookilerinin bir DOM belleğinin, Javascript namespace'inin bu kaynağa ait DOM'un olduğu ortamda nasıl yönetilip nasıl erişilebileceği idi.

Buna çözüm olarak yine Netscape mühendisleri browser tarafından yüklenen her bir kaynağın sınırlarını tespit edip bu sınırların birbirleriyle kurdukları ilişkileri bir kural ile yönetmeye kadar verdi ve SOP ortaya çıktı.

Yani SOP ile ; tarayıcıya yüklenen tüm kaynaklar bu kaynağa erişimde kullanılan protokol, URL ve erişilen PORT bileşiminden oluşan ve origin olarak tabir edilen bir dizge ile tanımlandı ve bu sayfalar aynı origine sahip ise birbirlerinin kaynaklarına erişebilecek hale geldi.

Örnek ile anlatım :

www.example.com 'a girdiğinizde, bu sayfadaki bir iframe'in www.badbank.com 'u yüklediğini varsayalım. Aynı tarayıcı üzerinden www.badbank.com 'a oturum açtı iseniz, bu oturuma ait tarayıcı belleğinde saklanan Cookie, www.badbank.com 'a yapılan istek ile beraber gönderilecek ve bu Cookie ile kimliğinizi doğrulayan www.badbank.com size ait olan bir içeriği yanıt olarak gönderecektir.

DOM nesnesi ile tüm döküman (document) nesnesinin özelliklerine ve olaylarına erişebildiğimize göre, aşağıdaki gibi bir kodun güvenliğiniz için oluşturulabileceği riskleri hayal edebiliyor musunuz?

```
<html>
<head>
<title>Welcome to Example.com</title>
</head>
<body>
<iframe name="bank_frame" src="http://www.badbank.com"></iframe>
<script>
   alert(frames.bank_frame.document.getElementById("balance").value);
</script>
</body>
</html>
```

```
<html>
<head>
<title>Welcome to Example.com</title>
</head>
<body>
<iframe name="bank_frame" src="http://www.badbank.com/sendmoney.php"></iframe>
<script>
frames.bank_frame.document.getElementById("recipient_account").value=1231;
frames.bank_frame.document.getElementById("fund").value=1231;
frames.bank_frame.document.getElementById("sendMoney").click();
</script>
</body>
</html>
```

Iframe vasıtasıyla www.badbank.com sizin geçerli oturumunuz ile yüklendi; fakat sizin rızanız olmayan bir isteği çalıştırmaya zorlandı.

PROTOKOL: http://

URL: www.example.com

PORT: 80

Bu üçünün birleşimi origin tanımını üretiyor.

Örnek verelim. Bir sitemiz var diyelim URL şu şekilde: http://www.example.com/dir/page.html

Bu site üzerinde aşağıda tanımlanan URL'lere yapacağımız çağrılara Same Origin Policy'nin izin verip vermeyeceğini kontrol edelim.

Compared URL	Outcome	Reason
http://www.example.com/dir/page2.html	Success	Same protocol, host and port
http://www.example.com/dir2/other.html	Success	Same protocol, host and port
http://username:password@**www.example.com**/dir2/other.html	Success	Same protocol, host and port
http://www.example.com:**81**/dir/other.html	Failure	Same protocol and host but different port
https://www.example.com/dir/other.html	Failure	Different protocol
http://**en.example.com**/dir/other.html	Failure	Different host
http://**example.com**/dir/other.html	Failure	Different host (exact match required)
http://**v2.www.example.com**/dir/other.html	Failure	Different host (exact match required)
http://www.example.com:**80**/dir/other.html	Depends	Port explicit. Depends on implementation in browser.

Neredeyse tüm modern browserlarda geçerli olan SOP ile web kaynakları birbirlerinin döküman içeriklerine, özelliklerine , yalnızca aynı protokol, aynı domain ve aynı port'a sahip oldukları, yani aynı origin'e sahip oldukları takdirde erişip, değiştirebilirler. Aksi bir durumda, döküman özelliklerine eriştirmek ve değiştirmek **browserlar tarafından** engellenmektedir.

SOP kuralları :

Her bir site; Cookie, DOM Storage, Javascript Namespace ve DOM gibi kaynaklara sahiptir.

Her bir sayfa, kendi originini URL'inden alır.

Script'ler hangi kaynaktan yükleniyor olursa olsunlar, kendilerini yükleyen sitenin kontekstinde ve yetki alanında çalışır.

Image gibi medya tipleri, pasif kaynaklardır. Yüklendikleri origin içerisindeki obje ve kaynaklarına erişimleri yoktur.

Bu dört ana fikri tüm olası durumlar için sıralarsak.

A originindeki bir site;

B kaynağındaki bir script 'i yükleyerek, kendi kontekstinde çalıştırılabilir.

B kaynağındaki bir script'in raw koduna, source koduna erişemez.

B kaynağındaki CSS'leri yükleyebilir.

B kaynağındaki CSS'lerin raw-text haline erişemez.

B kaynağındaki bir sayfayı iframe ile kendisine dahil edebilir.

B kaynağından yüklediği bu iframe'in DOM'una erişemez.

B kaynağındaki bir resim yükleyebilir.

B kaynağından yüklediği bu imajin bit'lerine erişemez.

B kaynağındaki bir videoyu çalıştırabilir.

B kaynağından yüklediği bu video'nun imajlarını capture edip, tekrar oluşturamaz.

Bu sayede web artık daha güvenli bir ortam sunuyor. Bugün webte çalışan her uygulamada(her tarayıcıda) SOP vardır.