OWASP
The Open Web Application Security Project

2012年5月

目录

来自编辑的序言

Deepak Subramanian

我们非常感谢大家对上一期新闻简报做出的积极响应。另外，新闻简报的投稿量也得到了很大提高。但是，我们依然希望大家对我们的新闻简报踊跃投稿。

在这里，我们为2012年7月出版的下一季度新闻简报招稿。

所投的稿件可以一次性递交一篇完整的文章，也可以是尚未完成的文章。

对于尚未完成的稿件，我们期望的时间表为：

1. 2012年6月15日——提交文章的摘要；

2. 2012年6月30日——提交文章的初版；

3. 2012年7月20日——提交文章的最终版。

如果你计划一次性提交一篇完整的文章，那么投稿截止日为2012年7月15日。

如果所投稿件在截止日以后提交，那么则被认为是9月新闻简报的投稿。

OWASP通讯简报希望刊登更多的研究类型文章。我们以极大的热情欢迎研究类型文章的投稿。

另外，我们感谢任何旨在将通讯简报做得更好的建议！

Email: deepak.subramanian@owasp.org

该中文版新闻简报由王颉[wangj@owasp.org.cn]编译完成。我们借此机会感谢王颉做出的贡献。

2011年OWASP AppSec Asia大会

Helen Gao, China AppSec 2011

2011年OWASP AppSec亚洲峰会于11月8日至11月11日在北京成功举行。此次峰会为期4天，其中包括两天会议和两天培训。来自十多个国家超过四百人出席了此次盛会。OWASP董事会成员Sebastien Deleersnyder通过一个号召参与并为计算机系统安全做出贡献的演讲拉开了大会的序幕。大会讨论的议题涵盖了许多应用安全的领域，包括：云安全、数据库安全、加密、安全的软件开发、RFID安全、以及减少XSS攻击和其他的威胁。丁丽萍博士、范渊博士、Cassio Goldschmidt、Tobias Gondrom、 Mano Paul、张炜博士、Meng-Chow Kang博士是此次大会的演讲者和培训师。在由OWASP全球分部委员会主席Tin Zaw领导的讨论会上，来自中国、韩国、马来西亚和印尼的领导人与来自欧美和南美分部的领导人交流了经验和看法。这是中国分部第三年举办大规模的OWASP活动。这也是第一次举行正式的产品展览会的活动，其中，14家厂商参加了展销。七家国内外的媒体对会议进行了报道。此次峰会也庆祝了OWASP的十周年庆典，以及OWASP组织，特别是在亚太地区的茁壮成长。在去年，中国分部的注册会员已增加了百分之四百，数量已超过了八百人。

会议现场

展览会

展览会

展览会

展览会

学生志愿者

会员委员会

2012年的主要目标是将我们的会员增加至少20%。我们将组织一些活动全球招募OWASP会员。我们将不停的寻找可以促使人们加入OWASP的方法。在过去，我们创建了一些针对于组织企业和个人的模型。随着时间的流逝，有的模型已经变成了多余的，有的模型则被证明是令人感到困惑或者不是有效的。我们将简化个人支持者、明确各地分部的支持者以及会议支持者。我们还将定义业内、高校以及政府支持者的角色和职责。

如果你认为该目标有点难以实现的话，那么，你是正确的。会员委员会目前是七个委员会中最小的一个。如果你考虑加入OWASP委员会，那么，会员委员会就是你能大展拳脚的那一个。如果你相信OWASP稳定的财政状况、如果你对来自个人和组织所做贡献的重要性而感到信服、如果你同意委员会应当更好的代表广大的参与人员，那么，请加入我们会员委员会并让它成为事实吧！

会员委员会和联系委员会已经成功举行了一次会议以寻求能延伸到主流媒体和科技媒体的新方法。在未来的几周里，我们将会连同各地分部的领导人和其他委员会为企业支持者会员的招募和个人支持者会员的招募建立计划。

你是否认为OWASP会员对于你个人或者你所在的企业组织重要？你个人或者你知道的某人是否进行过一次成功的会员招募？我们欢迎你的建议。会员委员会在美国东部时间每个月的第三个星期二中午12点举行电话会议。会议的日程安排将在会议以前发送到邮件列表里。为了参加会议，你不需要成为一个委员会成员。想加入邮件列表，仅需在OWASP的主页上点击“Mailing List”，然后选择“Global_membership_committee”。或者，直接访问“http://tinyurl.com/OWASPMembership”即可。你还可以直接发送邮件给helen.gao@owasp.org。

祝贺OWASP ZAP项目

全球项目委员会领导人 - Jason Li

OWASP新闻简报与全球项目委员会祝贺OWASP ZAP项目赢得2011年度Toolsmith工具奖。ZAP项目于2011年11月被刊登在ISSA期刊的Toolsmith文章中。Toolsmith在每个月中都重点介绍一个与安全相关的不同项目。ZAP是一个OWASP项目的很好例子—— 一个由充满激情的领导者所领导的帮助改善应用安全的开源项目。恭喜Simon Bennetts（又名Psiinon）做出的出色工作！如果你还没有浏览过OWASP ZAP项目，请阅读：https://www.owasp.org/index.php/OWASP_Zed_Attack_Proxy_Project

状态混淆

ASP.NET状态管理的安全性

Tim Kulp

试想一下，你收到一个包裹并被告知需要把它送去邮局。你尽职尽责地将包裹送到邮局，并把它交给了工作人员。工作人员看着你，问道：“你的包裹用的是泡泡包装纸还是小泡沫？”你想了一分钟.....你刚才才收到包裹时，没人告诉你它是如何包装的。你不知道包裹在递交到你手中以前发生的任何事。这就是HTTP的世界，一个无状态的协议，每一个发送于客户端和服务器之间的请求都与任何以前的请求相不连接。 HTTP不知道在递交请求以前发生了什么，只知道它有资料被送到了一个地址。作为Web开发人员，我们需要通过建立状态管理解决方案，以弥补HTTP的无状态性质。在这篇文章中，我们将用ASP.NET探讨各种状态管理解决方案以及各自的关切的安全焦点。当本文结束时，你将拥有对状态管理缺陷的很好知识，包括如何使用ASP.NET为你提供解决方案，如何保证你的状态数据安全。

特殊的挑战/可能产生的问题

开发人员时常会想到发生在客户端或者是服务器端的状态管理（通过HTML字段、cookies，等等）。但他们常常没有认识到，客户/服务器应用为状态管理提供了三种环境：

1.在客户端上，

2.在服务器上，

3.在客户端和服务器上

每一个环境都存在可能破坏你应用程序的安全漏洞。试图建立一个自定义的状态管理解决方案以解决所有三个环境的安全漏洞，是几乎不可能的挑战，即使是最有经验的工程师也可能陷入困惑。你可能会有这样的问题，我要如何保留在客户端和服务器之间的控制设置？或者是，我应该如何防止一个攻击者通过重放一个合法用户帐户上的恶意状态，从而发现漏洞威胁、应对措施和补偿控制。根据OWASP Top 10，最普遍且高风险的应用程序安全问题是“失效的认证和会话管理”。它可以直接涉及到开发人员尝试建立自定义的会话以及状态管理，但最终只形成自己的应用程序安全问题（OWASP Top 10，2011年）。自定义的会话/状态管理解决方案可能解决应用程序的迫切需要，但往往无法涵盖全部问题，比如：与身份认证相关联的会话/状态，当一个用户注销时，他们的会话信息被清除。最后，受项目的时间和预算限制，自定义状态管理解决方案可以导致比它们期望解决问题更多的安全问题。

幸运的是，许多拥有状态管理解决方案的开发框架已被广泛测试，并得到深入研究。ASP.NET提供了一个健康的状态管理解决方案。根据你的理解，在ASP.NET中的状态管理可以是复杂、混乱的，也可以是简单、可靠的。本文将帮助您了解视图状态和唾手可得的工具，以帮助你记录用户信息。通过探索每个状态管理工具，我们将检验它对应的漏洞以及相应的措施以保护你的系统。

客户端状态管理

在客户端状态管理中，客户端维护的状态信息随着每次请求发送到服务器端。图1（Northrup & Snell, 2010, 第121页）描绘了对于一个公司报告应用程序客户端的状态管理解决方案。用户订阅了两份报告，并自定义它们的主题为“SPRING”。每个请求被发送到服务器并得到处理。

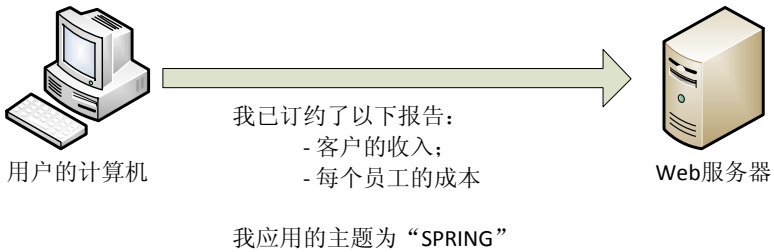

图1：客户端状态管理模型样本 (Northrup & Snell, 2010, 第121页)

使用客户端状态管理为您的应用程序提供了可扩展性。由于数据存储在客户端系统中，服务器的内存可替代数据存储从而进行处理运算。虽然这种解决方案可以提供很好的可伸缩性，但是你付出的代价是将状态数据暴露在客户端。这里，我们将介绍第一个对于所有的客户端状态管理解决方案常见的攻击：参数操纵。恶意用户可以修改存储在客户端的参数，从而滥用系统对于客户端的信任（Meier等人，2003年）。在图1中，如果一个恶意用户修改了客户端收到的报告，并且服务器端并没有验证用户对于资源的权限。恶意用户从而可以访问那些他们没有访问权限的资源，从而违反资源的保密性。“参数操纵”将是一个在检查由ASP.NET提供的客户端状态管理工具时，重复出现的问题。每个控制为这种攻击提供了各自的焦点和解决方案。

ASP.NET提供了一系统的客户端状态管理工具，其中包括：

•Query String参数

•隐藏字段

•Cookies

•视图状态

•状态控制

无论你使用的是什么工具，客户端状态管理的黄金法则是永远不要在客户端上存储敏感信息（Meier等人，2003年）。存储的信息，比如：为一个控制而存储字体信息是可以的，但是信用卡信息、个人健康记录、或任何其他被视为敏感的信息则应存储在服务器上。同样，不要在客户端上存储为作出安全决定所需的信息。如果应用程序完全依赖于存储在客户端上的参数，那么通过参数操纵，攻击者就可以提高它们的权限或执行未经授权的行为。

Query String 参数

通过使用HTTP GET参数， ASP.NET可以将数据从客户端通过Query String发送到服务器。举一个例子，在下面的URL中，我可以把我的产品ID发送到产品信息网页：

http://some-e-com-site.com/product-details.aspx?id=9

然后，product-details.aspx网页可以使用下面的代码装载id参数：

```
int prodId = Convert.ToInt32(Request.QueryString[“id”]);
```

Query String参数对于少量的数据而言，是非常理想的，因为许多浏览器将URL的长度限制在2100 (Northrup & Snell, 2010)。 传递标识，缩写，等等。Query String参数的另一个用途是，允许用户为具体要求标记请求。对于上面的例子，一个用户可以发送URL给朋友，以直接查看id为9的产品，或者用户可以迅速返回到这个地址再次看到产品的详细信息。

恶意用户可以很容易通过他们的浏览器（或HTTP请求传输工具）修改查询字符串参数。Query String参数是第一个探讨的状态管理解决方案，因为它们是参数操纵最简单的例子。当使用Query String参数时，有以下几个问题：

1.参数的边界值是多少？最大值是多少？最小值是多少？

2.如果有人传递的值不是我所期待的该怎么办？

这些是标准的输入验证问题。第一个代表边界值测试。对于我们的URL例子：

http://some-e-com-site.com/product-details.aspx?id=9

如果id=-1会发生什么呢？系统将会崩溃，什么都找不到，或者是显示一条消息显示 “id”是一个无法识别的值？通过使用像Fiddler（http://fiddler2.org）或Hackbar（Johan Adriaans开发的一个非常好的Firefox扩展）这样的工具，Query String参数让操纵参数变得非常简单。在数据点周围建立可接受的边界值，并确保传入的数据符合这些边界。举一个例子，如果有的e-com-site.com只有10个产品（编号从1至10），你的应用程序并就不需要支持大于10或小于1的ID。

上述的第二个问题是一个等价划分的例子。该测试技术将输入分组成为数据块，以尝试减少需要测试的数量。比如，一个部分可能是字母字符。如果我传递id=A，那么系统就会因为字母字符而崩溃。我不需要测试每个字母字符以验证系统是否会崩溃。划分的数据可以在已知数据范围以外（比如：当我们的id只从1到10时，id=11或者id=0）。通过输入其他区域的数据以作为一个安全测试，我们以确保系统维持期待的行为，并且不泄露信息，或者显示错误信息的黄屏（YSoD）。

为了保护我们的系统，我们可以在实际使用它以前，通过各种检查前的数据加强防御。具体到ASP.NET和C＃（或者VB.NET），你可以使用tryparse方法验证提供的数据，以保证数据类型的转换。如果提供的值可转化为预期的数据类型（比如：int），则该方法返回true。 和parse方法不同，如果值无法转换， tryparse方法不会抛出异常。在C#中，许多数据类型（比如：布尔、日期时间、十进制，等等）都支持tryparse方法，以允许开发人员检查数据类型并维护他们的应用程序流程没有异常。这里有一个使用tryparse的简单数据验证：

```
if) Request.QueryString]«id»].Length < 3)

{

        int prodId;0 =

        if!) int.TryParse(Request.QueryString[“id”], out prodId((

                displayMessageToScreen(“Product Id must be a recognized value.”);


      if) prodId 1 > || prodId(10 <

            displayMessageToScreen(“Product Id is not in a valid range.”);

}
```

此代码首先验证Request.QueryString[“id”]值的长度不超过3个字符。这将避免接收一个长度大于两位的数字。接下来，我们检查Request.QueryString[“id”]可以为C#强制转换为int的数据类型（这是一个Int32类型）。如果该值可以被转换int类型，那么产品id将被设置到Request.QueryString[“id”]。如果该值不能转换，displayMessageToScreen方法被执行以显示一个友好的错误信息给用户（从而避免了可能的

YSoD和错误抛出，因为从内存的角度而言，抛出错误异常的代价很大，而YSoD则让你的网站看起来像是坏掉了）。下一步，我们确认值处于预计的边界范围内，如果该值小于1或大于10，则再一次显示错误。这是一个简单的例子，但是，传达的想法是如何使用tryparse方法检查一个值是否可以转换，然后用简单的输入验证，以确保边界值范围。

隐藏字段

很久以前，我和一个开发人员聊天，他说他们的数据很安全，因为它们存储在隐藏字段之中。由于用户无法看到它们，那么这些数据是安全的。不幸的是，这个观点是极其错误的。数据对于不窥探、非恶意、不好奇的用户是安全的，但永远记住，软件只是一种工具，人们喜欢用他们的工具捣鼓。为此，认为“隐藏字段提供了安全”就同意了“不公开，即安全”的谬论。

隐藏字段是有“隐藏”类型的HTML input标签。这可防止标签被作为网页布局的一部分呈现出来，但是，通过查看HTML页面的源代码则可以很容易的发现。 ASP.NET隐藏字段的创建使用了以输入类型为“hidden”标签的HiddenField控件：

```
<asp:HiddenField ID="hdn1" runat="server" Value="Some String Value" />
```

表现为：

```
<input type="hidden" id="hdn1" name="hdn1" value="Some String Value" />
```

隐藏自动的好处可以在向网页（比如：GUID）传递非用户友好的数据，或者为AJAX应用使用一个数据容器以在Postback时发送给服务器时体现出来。隐藏字段可以通过JavaScript访问（读/写），使他们成为理想的桥梁，为一个允许AJAX的网页从客户端传递数据。

除了知道参数操纵攻击以外，开发人员还需要知道隐藏字段不为存储的数据提供保护。只需通过查看页面源代码，字段值就可以暴露。使用HTML隐藏字段需要开发人员责任，以确保提供适当的数据管理。一些开发人员已经建立了非常有趣的客户端加密系统（由Daniel Griesser开发的jCrypt非常有趣且易于实现），以用来加密隐藏字段中的数据。你可以建立一些正则表达式用于检查数据，但是，你自己做决定吧！

ASP.NET的优点之一是验证控件。通过使用RequiredField控件，你可以确保ComapareValiadtor提供或使用的符合其他控制的参数。不幸的是，ASP.NET验证控件不使用HiddenField控件。所有隐藏字段的验证需要手动完成。当处理隐藏字段时，确保你使用的是正确的输入验证，以保持在系统传递时数据是安全的。避免基于隐藏字段的任何值来做出有关安全的决定，并始终复制服务器上针对隐藏字段数据的任何安全检查。

Cookies

Cookies已经在网络上出现很长一段时间了。它们广泛存在于网页之中，因为网站利用它们存储广告活动的信息、客户ID、喜欢的颜色，从而使各种各样的网页成为可能。不同于Query String参数和隐藏字段，Cookies在页面被关闭后依然可以长时间存在。一个cookie的存在时间在产生过程中被定义。

```
HttpCookie cookie = new HttpCookie("roles");

cookie.Value = "Access Maps, Access Reports, Access Reports";

cookie.Expires = DateTime.Now.AddMinutes(30);

Response.Cookies.Add(cookie);
```

在上面代码的第三行显示，从现在（日期和时间）开始，cookie将在30分钟以后过期。Cookie的值可以是任何字符串（可能为：一个序列化的XML对象、JSON对象、或者是代表其他某些数据类型的字符串）。作为一个可以存储任何东西的cookie，你只会被你的想象力和一个非常小的文件大小（4KB）所局限。 Cookies作为一个小的文本文件保存在客户端系统，它是网站唯一可用来创建的。 HTTP头将cookie从客户端传送到服务器，使他们的数据提供给服务器端的C＃以及客户端的JavaScript。

和所有的客户端状态管理解决方案一样，cookies容易受到参数操纵攻击。由于数据只存储在一个文本文件，在默认情况下是没有加密的，因此对于下一次连接，其内容可被看到、可被操纵、可被保存。凭借其在HTTP头中传输，cookie的值可以通过像Fiddler或者是OWASP的ZAP工具在传递过程中篡改。参数操纵的一个具体表现是操纵一个角色缓存。有时，开发人员将缓存一组角色列表，以减少为确定用户角色而每一次系统需要授权访问数据库。虽然从表面上看，这似乎是一个很好的缓存解决方案，对cookie的修改可以改变用户的角色。举一个例子，在我们前面的cookie中，想象把cookie角色的值修改为“Admin”、“Administrator”或者其他管理员访问的同义词。如果应用识别了其中一个角色，但不验证其对应值，用户就可以提升其系统的权限。

Cookie面临的另外一个漏洞是Cookie劫持。这是一个恶意用户窃取合法用户的cookie。Cookie劫持实际上是一个跨站点脚本（XSS）攻击。只需几行恶意代码，你就可以获得cookie的内容：

```
$(document).ready(function () {

$("#btn").click(function (e) {

            _stealTheCookie(document.cookie);

      });

});


function _stealTheCookie(val) {
```

```
$("#output").html(val);
}
```

虽然cookie劫持可能获得类似用户喜欢的颜色这样无关紧的简单信息，但是，如果开发人员在cookie中存储了敏感信息时，则会很致命。如前所述，绝对不要在任何客户端状态管理解决方案中存放敏感信息。即使是在加密数据时，客户端机器上存储的东西任处于一个充满恶意的环境，并会被公开曝光。

最后，让我们再来看一段盗取cookie的恶意代码。它显示了为浏览器将值设为HTML的cookie。我们知道cookie将会存储在系统中：

```
roles=Access Maps, Access Reports, Access Reports
```

如果我将cookie的值修改为：

```
cookie.Value = "<iframe src='malsite.com'></iframe>";
```

这会为浏览器提供一个iframe，并通过一些有创意的CSS，可以导致一个非常有说服力的网页更换。当用户认为他们是在和合法网站互动时，他们其实是在浏览malsite.com。再次提醒，你必须总是在你的应用程序中验证输入并净化传递给用户的数据。

视图状态

在ASP.NET中使用的所有状态管理解决方案中，视图状态可能是最普遍的，但是也是了解最少的。在默认情况下，每一个ASP.NET网页都带有一个名为__VIEWSTATE的隐藏字段。此字段包含一个64位的编码值，以代表网页上所有控件的状态。根据ASP.NET网页上控制的复杂度，视图状态可以非常小或非常大。这里有一个__VIEWSTATE隐藏字段的例子：

```
<input type="hidden" name="__VIEWSTATE" id="__VIEWSTATE"
value="/wEPDwULLTEyMTE0MTI3OTJkZPmGU1lm0YsgU53Toeaoeeoajf92zZLCje8JDdoOSpwDe1dD/" />
```

许多开发人员都看到了 value 的值，但是由于缺乏关于视图状态的知识，从而假定它是一个加密值。和其他的隐藏字段一样，__VIEWSTATE 不提供任何默认的保护，以防止用户读到内容信息。视图状态的目的是为了存放关于网页控制的信息，比如在回发时由用户提供的值。这段代码降低了开发人员编写传递给服务器所有字段的重复性，而它比起一个安全控制，更像是一个提供便利的工具。在许多代码样本中，我所看到的自定义控件（ascx 控件或者是自定义控件库）使用视图状态以为所有类型的数据包括全部的数据集提供存放的地方。客户端状态管理的黄金法则是使用视图状态，并在建立 ASP.NET 控制时一定要记住：“永远不用在客户端存储敏感信息”（Meier 等，2003 年）。当你遇到别人的代码没有遵守黄金法则时，ASP.NET 可以通过把 Page.ViewStateEncryptionMode 为“True”来对视图状态加密。通过使用该属性，ASP.NET 将处理视图状态数据的加密和解密。和所有的加密一样，数据的复杂度可能影响到性能，因此，只有当你无法保存视图状态的敏感数据时才使用该选项。

当数据没有加密时，应对数据签名，以确保视图状态没有在客户端上修改。默认情况下，ASP.NET 使用了采用 MachineKey 算法的机器身份认证代码（MAC）对视图状态进行签名，并提供在 web.config 文档中。如果在服务器上计算获得的 MAC 值与来自客户端的值不匹配，则抛出一个异常以防止应用继续处理视图状态。当 MAC 防止了对视图状态值的修改时但是它不能组织一个恶意用户在稍后时间再次提交视图状态。这也被称为视图状态重放攻击（Baier，2006 年）。通过捕获一个有效的视图状态，攻击者可以为其他用户制作一个恶意的视图状态。制作的视图状态可以通过其他的攻击，如：跨站点脚本攻击（XSS [https://www.owasp.org/index.php/Top_10_2010-A2]）、或跨站点请求伪造攻击（CSRF [HTTPS://www.owasp.org/index.php/Top_10_2010-A5]）。使用视图状态重放攻击，受害人可以向一个 web 应用提交看起来像是合法数据流的恶意请求。幸运的是，通过使用 Page.ViewStateUserKey 属性可以很容易的阻止该攻击。所有的 ASP.NET 网页中有一个名为 ViewStateUserKey 的属性，它允许开发人员在视图状态中放置一个独特的种子以绑定一段独特的数据。通常情况下，会话 ID（我们将在服务器状态管理章节深入研究 Session 对象）被用来绑定一个特定的用户会话视图状态。不论你是否使用会话 ID、用户名或二者的哈希值，ViewStateUserKey 的关键是使用一些对服务器可用、且对用户独一无二的数据。使用下面的代码行，你可以为用户会话 ID 指定 ViewStateUserKey：

```
Page.ViewStateUserKey = Session.SessionID;
```

稍后，我们将研究会话ID如何可以被劫持，以导致攻击规避这种控制。考虑到当你在创建ViewStateUserKey的时候，你的应用所需要的安全等级。会话劫持是一个攻击吗？你可以使用其他数据吗，比如：用户名或者用户个人资料中的一个值（我们稍后将研究个人资料）？

开发人员常犯的另一个错误，是假设视图状态是安全的。当在处理值和内容时，视图状态应当被对待为另一个输入字段，就象一个文本字段一样。当视图状态MAC可以提供一个保护等级以抵抗外部的恶意用户时，作为开发人员，我们需要建立我们的组件以保护来自内部的威胁。一个心怀不满的开发者可以构建恶意组件注入到Web应用程序中的漏洞。通过简单处理视图状态的内容，你的Web应用程序可以为愤怒的开发人员攻击他们的雇主提供了机会。始终验证应用程序的输入并净化任何被返回给用户的信息。

控制状态

“视图状态”的小弟弟，“控制状态”，是不能被禁用的视图状态。使用Page.ViewStateEnabled，开发人员可以关掉页面或整个应用程序（在配置级别）视图状态。由开发人员使用的控制状态，在视图状态被禁用的情况下坚持值。控件状态存储在__VIEWSTATE隐藏字段，因此，从安全角度来看，它是一个视图状态的扩展。通过适当的保护视图状态和输入验证，状态控制将得到保护。

当客户端提供了大量的状态管理选项时，服务器提供了不同的选择和自己特有的挑战。在服务器端状态管理中，取决于所使用的技术，有的信息被提供以识别用户的服务器。服务器存储和管理状态信息。该解决方案避免了客户端状态管理中参数操纵，但是它也为你的应用带来的挑战和机会。通过在ASP.NET中使用以下对象以在服务器中维护状态。

•应用程序

•对话

•资料

•视图状态（视图状态可以存在于服务器中）

每个对象对带有自己的安全包，通常与访问和范围相关。对于访问，你需要知道你的应用程序将如何访问不同的状态对象。范围是指正常范围的数据，让你不提供范围之外的内容。我们将研究这些方面，因为它们适用于各种服务器端状态的对象。

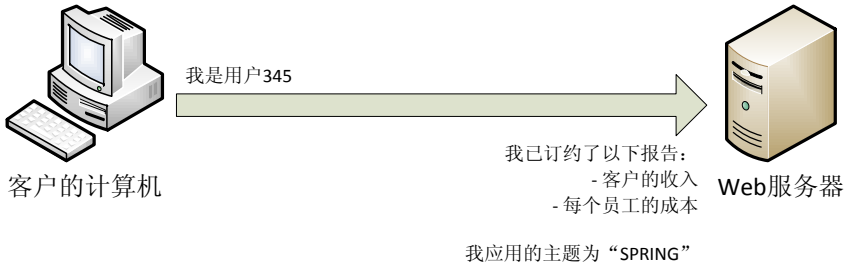

图 2

应用程序

应用程序用于存储一个ASP.NET应用所有用户需要的信息。数据存储在内存中以允许快速存储和检索。这个对象对于存储那些不随用户而改变（如：默认的ID值），且少量的数据是非常理想的。 应用程序的对象是一对当IIS Web应用程序启动时实例化和当应用程序停止时失去的关键值。

应用程序对象的使用，应仅限于少量的全局数据。数据范围不当，可能会导致信息泄露的漏洞。当用户把具体信息存储在应用程序对象中时（应用对象不应该专门针对用户的具体信息），对Web应用程序的任何用户都是可用的。虽然这是一个由于使用不当而导致安全漏洞对象的例子，但它是一个很好的例子，以说明一个简单的错误可能会导致当其他人使用该应用程序时使用你的用户数据。当使用应用对象时，确保所存储的数据不是一个特定用户的。

会话

会话对象是一对与个人用户相关联的关键值。和应用程序对象不同，会话不是在全局范围内的。用户通过使用传递给客户端的SessionID值来确定服务器。此ID在每个请求中进行传送，然后由服务器为用户确定什么（如果有的话）数据存储在会话数据存储中。会话请求被发送到服务器内指定的会话有效期是长期活跃。这允许会话在指定的一段时间后过期。

默认情况下，SessionID存储在客户端系统上的Cookie之中。ASP.NET为SessionID在客户端和服务器的通信支持另一种选择：通过URL。这使SessionID的作为URL（使用URL重写）的一部分。通过在web.config文件中添加以下元素，你可以配置你的ASP.NET应用程序通过URL传递SessionID的：

```
<sessionState cookieless=”true”/>
```

这将使一个URL看起来象下面的样子：

http://localhost /**(S(tmuwrs2ubkjgnxi4ulrznncy))**/default.aspx

注意到“tmuwrs2ubkjgnxi4ulrznncy”是SessionID。ASP.NET将会通过应用中将会话ID自动添加到每个URL中。这是一个很好的解决方案，虽然客户端系统不需要接受cookies，但是导致了大量的安全问题。

一旦SessionID被提交到客户端计算机，就很容易遭受参数操纵攻击。当SessionID被随机生成并不被容易猜到时（MSDN，2011年），它可以被攻击者通过捕获明文传输的SessionID而修改为另一个合法的Session ID。通过使用网络监控系统，比如：WireShark，攻击者可以在网络中搜集明文的SessionID。当搜集到了SessionID以后，攻击者可以通过修改SessionID以获得其他用户存储在会话中的敏感数据。为了避免暴露SessionID，应当使用如SSL/TLS加密算法以避免通信中的明文传输。

另外一个由于使用无Cookie会话而导致的安全问题是SessionID的回收。默认情况下，如果一个SessionID通过URL ASP.NET提交的话，将与所提供的ID创建一个会话。这可能会导致两个用户有相同的SessionID，并导致系统中信息泄露的漏洞。通过电子邮件或者搜索引擎发送嵌入SessionID的URL，另一个用户可以获得其他用户在会话中存储的任何数据。为了复制此漏洞，打开一个使用无Cookie会话网站（或者自己创建[http://seccode.blogspot.com/2012/03/cookieless-sessions-with-aspnet.html]）。将URL复制到另一个浏览器（实质上是创建一个新的会

话）并装载网页。请注意，任何预设的会话变量会跟随你出现在新浏览器中。

ASP.NET会话状态提供了众多的挑战，以确保应用程序的安全性，但是没有任何一个办法是100％安全的，我们可以添加防护层以使对会话的攻击更加困难。第一也是最重要的是：管理。当用户注销或离开（如果可能的话）应用程序时，确保你结束会话（使用Session.Abandon（）方法）。将abandon方法与regenerateExpiredSessionId配置相结合，将大大减少通过共享一个会话ID而导致的会话劫持。 sessionState配置的regenerateExpiredSessionId属性被默认设置为“true”。此设置可以确保当客户端尝试使用旧的会话ID时，一个新ID被生成以取代旧的ID。使用放弃未使用的会话设置将减少在会话ID处于使用状态下的攻击。虽然不是一套绝对的安全方法，但是将配置和最佳实践相结合的方法可以减少可能的攻击面（暂时看来）。

下一步，考虑对特定个人用户签署含有信息的会话ID。通过同时使用机器验证代码（MAC），你可以添加一个额外的验证层，以为用户验证会话ID的值是真实的。在《挫败会话劫持》[http://technet.microsoft.com/en-us/query/cc300500]一文中，Jeff Prosise通过在会话ID中添加由用户代理、用户IP地址和会话ID组成的MAC探讨加入组成的会话ID，以验证信息的真实性。需注意的是，这些值都可能是伪造的。但是如果使用一个MAC作为另外一个校验收到的会话信息，可以使一个会话更加不容易被劫持。在文章中，Jeff使用了一个ASP.NET模块来捕获所有传入的请求，以验证所提供的会话MAC。如果MAC是无效的，应用程序记录尝试进行的企图劫持并拒绝其访问。

最后，考虑会话ID值的安全传输（通过SSL）。通过URL或cookie暴露会话ID，可导致上面讨论的无数劫持漏洞。通过加密传输的数据，可以减少在传输过程中捕获到会话ID的可能性。

配置文件

到目前为止，我们研究的所有状态管理解决方案都是暂时的。会话状态的过期，客户端的方法依赖于一个指定的时间寿命，甚至应用程序状态可以因为IIS重启而失去。配置文件提供了一个持久的状态管理工具，以允许用户在你的应用程序中存储状态信息，以等待用户的再次访问。

配置文件与个人用户相关联，并根据配置文件进行存储。在默认情况下，ASP.NET使用了SqlProfileProvider将配置文件信息存储到Microsoft SQL Server或SQL Express数据库。 ASP.NET Provider模型的检验超出了本文的讨论范围，更多信息可以在MSDN中找到（MSDN [http://msdn.microsoft.com/en-us/library/014bec1k.aspx]）。Provider模型的核心概念是能够轻松地管理和配置Web应用程序中常用的重用功能。对SqlProfileProvider而言，这意味着定义如何在SQL Server或SQL Express数据库中存储资料信息。你可以自定义配置文件，以将数据存储在Oracle、SQLLite、XML或者其他你想要的数据格式。这里第一次介绍到了在ASP.NET配置文件中的漏洞：不恰当的建立Provider。雄心勃勃的开发人员当听说Provider模型以后，并想迅速建立他们自己的模型。就像破损的身份验证方案一样，一个破碎的Provider模型可以导致你的系统中出现许多漏洞。使用ASP.NET已有的Provider模型可以降低一个不安全切自定义开发出来的Provider模型的风险。如果你需要建立你自己的模型，仔细审核编写的代码，以确保它遵循了安全代码的最佳实例，避免已知的攻击模式（如：SQL注入）并确保安全状态。

配置文件使用了一种极其灵活的实施方案。在web.config文件中，开发人员指定了通过添加元素到Profile.Properties对象中，哪些文件元素是可用的：

```
<profile>
   <providers>
      <clear/>
         <add …/>
   </providers>
>    properties>
      <add name=”FavoriteColor” allowAnonymous=”false” type=”System.String” defaultValue=”Blue”/>
    </properties>
</profile>
```

在这个例子中，配置文件对象将有一个名为“FavoriteColor”的字符串属性，其默认值得是“蓝色”，且对匿名用户不可用（即用户必须通过身份验证访问此配置文件属性）。对配置文件属性赋值是非常容易的：

```
Profile.FavoriteColor = TextBox1.Text;
Profile.Save();
```

这就将FavoriteColor设置为TextBox1控件（ASP.NET文本框）提供的Text属性。保存以后，会将配置文件写入到数据资料（如在Provider中定义的那样）之中，以供以后使用。和配置文件工作可以很简单，你必须记住去验证它的数据。配置文件可以在应用程序中为存储的有害输入信息提供用攻击层面。要始终记住在服务器端验证用户的输入。

有些网站持续使用配置文件作用于应用程序。在这个例子中，FavoriteColor可能被用来定义一个CSS字符串，以修改用户进入网站时的背景。为了减轻网络流量，有时配置文件会被作为一个cookie存储在客户端。虽然FavoriteColor不暴露关键信息，其他常见的配置文件数据节点，比如：姓、名、出生日期等，都可能会暴露用户在客户端上的数据。当使用客户端缓存配置文件的数据时，需留意什么数据被存储以及你是如何存储它们的。默认情况下，配置文件保留在服务器和数据库中（通过其他一些ASP.NET控件显示）。通过使客户端远离敏感数据，是一个常见状态管理的最佳实践。

是的，视图状态可以在服务器上保存。 ASP.NET存储视图状态是由继承一个名为PageStatePersister的抽象类定义的。该类由诸如用于在网页上存放视图状态的HiddenFieldPageStatePersister类的主机使用。默认的PageStatePersister是HiddenFieldPageStatePersister，将视图状态作为隐藏字段在客户端上使用base-64编码输出。其他的PageStatePersister包括SessionPageStatePersister存储在Session对象中的视图状态。MSDN 提供了一篇关于如何使用 System.IO.Stream 来建立 PageStatePersister 的文章[http://msdn.microsoft.com/en-us/library/system.web.ui.pagestatepersister.aspx。建立自己的Persister程序，为你的代码带来了在自定义Provider时相同的安全问题。错误的假设和不安全的编码将使你的应用程序非常脆弱，并可能导致你的视图状态暴露出来。当建立一个自定义的Persister时，通过经常检讨代码，以确保最佳实践得到了使用，并且当你尝试将视图状态从客户端关闭时，你不会错误的创建一个漏洞。

你可以想像，从客户端删除视图状态可以为系统的安全性和性能带来很多好处。从安全角度来看，你删除客户端的视图状态，可以减少参数操纵和视图状态重放攻击。通过从客户端删除视图状态，你删除了攻击对存储数据的控制。当考虑到在什么地方存储视图状态信息时，需要考虑范围、访问和持久性。对于范围，在应用程序对象中将视图状态作为一个值来存储，以使每个人都使用该应用程序范围的数据。这显然是一个坏主意！因为它将同一个视图状态为许多用户打开。你将如何保存视图状态，以使用户可以访问它并使重放攻击是不可能的吗？“访问”有如下类似的思路，只有拥有访问权限的用户才能访问视图状态。如果你将所有的视图状态信息写入到一个文本文件，那么任何知道该地址的人，你的视图状态数据都会暴露给他。你的解决方案只为拥有视图状态的用户授予访问权限吗？最后，你必须考虑你要将视图状态信息在你的系统中保存多久。如果你坚持将视图状态保存到数据库中，那么有一个清洁过程以消除旧的视图状态吗？你将如何确保只有可用的视图状态信息可用来处理你的解决方案？回答这些问题可以帮助你建立一个安全自定义的PageStatePersister对象，并避免在执行中因可能存在的缺陷而导致意想不到的信息暴露。

结束语

ASP.NET提供了许多状态管理工具，每一个都有其特定的目的和用途。在拥有这么多选择的情况下，开发人员可能在使用时变得困惑。使用了错误的状态管理工具，可能导致HTTP运行恶意输入到你的服务器或导致攻击者窃取用户的数据。建设安全的状态管理，要求了解每个工具为应用程序带来的挑战与机遇。ASP.NET为你做出了有关状态管理的大部分工作。作为一名开发人员，你现在必须将这些拼凑在一起，锁定下来，以保证用户的数据安全。

在这篇文章中，我们探索了许多ASP.NET状态管理工具和各自的安全焦点。我们专注于客户端和服务器端的状态管理，它们的优点和弱点，以及如何将你的防护分层以允许各自的工作。通过使用实例，并提出呈现在文章中的问题，你可以为你的下一个ASP.NET应用程序建立一个强大和更安全的状态管理系统。

参考文献

Baier, D. (2006). *Developing More-Secure Microsoft ASP.NET 2.0 Applications.* Redmond, WA: Microsoft Press.

Ballad, T., & Ballad, W. (2009). *Securing PHP Web Applications.* Boston, MA: Addison Wesley.

Hickson, I. (2011, 11 28). *Web Storage*. Retrieved from W3C: http://dev.w3.org/html5/webstorage/

Howard, M., & Lipner, S. (2003). *Writing Secure Code.* Redmond, WA: Microsoft Press.

inferno...@gmail.com. (2010, 1 30). *Issue 33876: Security: LocalStorage Cross Domain Denial of Service Attack*. Retrieved from chromium project: http://code.google.com/p/chromium/issues/detail?id=33876

Meier, J., Mackman, A., Vasireddy, S., Dunner, M., Escamilla, R., & Murukan, A. (2003). *Improving Web Application Security: Threats and Countermeasures.* Redmond, WA: Microsoft Press.

Microsoft. (2005, 1 1). *Control State vs. View State Example*. Retrieved from Microsoft Developer Network (MSDN): http://msdn.microsoft.com/en-us/library/1whwt1k7.aspx

Microsoft. (2010, 6 22). *ASP.NET Session State Overview*. Retrieved from Microsoft Developer Network: http://msdn.microsoft.com/en-us/library/ms178581.aspx

Microsoft. (2011, 9 8). *ASP.NET Cookies Overview*. Retrieved from Microsoft Developer Network: http://msdn.microsoft.com/en-us/library/ms178194.aspx

Microsoft. (2011, 9 5). *ASP.NET State Management Overview*. Retrieved from Microsoft Developer Network (MSDN): http://msdn.microsoft.com/en-us/library/75x4ha6s.aspx

Mitchell, S. (2004, 5 1). *Understanding ASP.NET View State*. Retrieved from Microsoft Developer Network (MSDN): http://msdn.microsoft.com/en-us/library/ms972976.aspx

Mozilla. (2011, 11 26). *DOM Storage*. Retrieved from Mozilla Developer Network: https://developer.mozilla.org/en/DOM/Storage

MSDN. (2011, 12 31). *HttpSessionState.SessionID Property*. Retrieved from MSDN: Microsoft Developer Network: http://msdn.microsoft.com/en-us/library/system.web.sessionstate.httpsessionstate.sessionid.aspx

Northrup, T., & Snell, M. (2010). *Web Application Development with Microsoft .NET Framework 4.0.* Redmond, WA: Microsoft Press.

OWASP Top Ten. (2010). *OWASP Top 10 - 2010.* Internet: OWASP.

OWASP Top Ten Team. (2011, 12 15). *Top 10 2010-A3-Broken Authentication and Session Management*. Retrieved from OWASP: The Open Web Application Security Project: https://www.owasp.org/index.php/Top_10_2010-A3

http://msdn.microsoft.com/en-us/library/ms178594.aspx

有关本文的任何疑问，请联系：

Tim Kulp timkulp@live.com

如果您希望阅读本文修正后的版本，请联系：

Deepak Subramanian deepak.subramanian@owasp.org
Kate Hartmann kate.hartmann@owasp.org

2012年项目重启

OWASP项目重启的初步计划？

OWASP需要恢复、振兴并更新一些项目。我们需要让软件开发社区更多的了解到我们付出的努力，并证明解决方案和指导基础库的设计旨在帮助安全的应用程序开发生命周期。
初步计划的提案在下面的链接中：

项目重启计划提案

项目领导人： Eoin Keary
通过了计划的团队： Jim Manico, Rahim Jina, Tom Brennan

为此，我们准备了预算以资助各种与项目有关的活动。这一举措的预期成果是提供一些高质量的材料，用来在未来几年中，对软件开发人员和测试人员提供支持。

当前提交的项目

OWASP Application Security Guide For CISOs

OWASP Development Guide

OWASP Testing Guide – 达成协议，等待计划的提交。

Zed Attack Proxy

OWASP Cheat Sheets

OWASP AppSensor

OWASP Mobile Project

关键日期

提案提交截止日：2012年7月30日

提案的第一轮筛选：2012年6月15日

提案的第二轮筛选：2012年8月10日

活动类型

第1类：指南或工具的更新、重写和完全。
该类型旨在为了提供高质量的发布成果而需要更新、扩展、重写现有的和全新的工具或指南。

例如：

1. 基于小项目的峰会：为发布一个项目的新版本而参加国际研讨会的相关花费。
2. 为贡献者所付出的时间和努力而支付费用。
3. 为由个人开发编写的用户指南（技术类编写）而支付费用。

第2类：市场营销、培训、宣传、增加使用。
现有的优秀工具和指南可使用第2类活动以帮助建立有关项目的意识，并宣传促进对项目成果的使用。

例如：

1. 协助与项目营销的相关费用。
2. 促进OWASP项目的培训和宣传活动的花费。

为帮助保护一个现有的或将来的软件应用而捐赠

Donate

我可以申请这个重启项目吗？

如果您是一名OWASP会员，您绝对可以！
如果您觉得你的项目已经就绪，或者有潜力，您就可以申请这个重启项目。

资金是如何使用的？

第1种：资金可用于作为重启项目所需要的交通旅行、参加小型峰会等活动。而开发活动，则可以在项目达到50%或者100%的里程碑时向项目贡献者支付。

里程碑需要在项目重启初始时达成一致。

当工作按时达到50%的里程碑时，项目应当被GPC的一名委员和另外一名任命的OWASP审核人员（通常为一名OWASP领导人）审核。

第2种： 资金在被需要时提供。被赞助的项目需要在项目重启初始时达成一致。

需要报账的发票要直接送给委员会用于资金支付。

我应当如何申请呢？提交一份包含以下信息的提议草案：

1. 项目名称和描述。包括重启项目的领导人和所有成员；
2. 重启项目的类型（第1类或者第2类）；
3. 重启的目的；
4. 有关50%里程碑和100%里程碑的时间表，以及建议的里程碑审核人员（通常是OWASP领导人或者其他业内专家）；
5. 所需的预算，已经您将如何使用。

您想支持初始活动吗？或者了解更多信息吗？请联系：Eoin Keary

OWASP Podcast

主持：Jim Manico

OWASP Podcast系列由Jim Manico先生主持，并以采访各种各样的安全专家为特色。这周我们采访Troy Hunt，他是一位关于.Net安全的Microsoft MVP。

Podcast链接：https://www.owasp.org/download/jmanico/owasp_podcast_91.mp3

OWASP TOP和Hacking-Lab

Martin Knobloch

介绍

OWASP全球教育委员会（GEC）和Hacking-Lab正着手一个联合的教育项目：Academy Portal和Hacking-Lab的远程安全实验室。被动的学习方法通常被认为仅会达到一个较低的水平，而一个互动的学习环境将允许学习者达到一个较高的水平（i）。

OWASP Academy Portal https://www.owasp.org/index.php/OWASP_Academy_Portal_Project

什么时候开始的…

自2011年在Minneapolis的AppSec US大会发布以来，该门户网站已有了超过6000名来自世界各地的用户，

并且有超过1072名个人签署了开放的OWASP Top 10挑战。

积分系统

目前，一个别名为“bashrc”的用户在OWASP Top 10事件的积分系统中领先。在最近两个月，167名用户已经成功解决了OWASP挑战。

OWASP GEC团队不分白天黑夜的检查提交的解决方案。这些工作是由以下关键个人付出了努力：Martin Knobloch，Cecil Su，Steven van der Baan和Zaki Akhmad.

OWASP在线竞赛

OWASP正计划添加额外的挑战。感谢希腊的Hackademics项目，额外的挑战现已就绪，计划于2012年在OWASP的在线安全竞赛中使用。获胜者将获得一张参加任何一个OWASP国际会议的票。

整合WebGoat

通过志愿者的努力，WebGoat在前几周已整合到Hacking-Lab架构中。非常感谢来自赫尔辛基的Nicolas Hochart！主要的工作已经完成，我们在发布给公众以前正进行质量保证处理。Hackademics和WebGoat项目将引入超过20个新的和开放的挑战，以使每个人都可以获得一些实际经验。

高级Web安全

OWASP TOP 10是唯一一个重点关注的领域。许多额外的、关键的安全方面需要额外的关于。针对一些最近媒体上出现的讨论做出的回应，OWASP现在有了针对Apache Struts2安全漏洞和大众普遍不知道的XML外部实体攻击（XXE）的额外安全挑战。

Apache Struts2 教程：http://media.hacking-lab.com/movies/struts2/

LiveCD

不要犹豫，现在就着手开始研究！加入OWASP Top 10挑战是很容易的。登录一个Hacking-Lab的帐号，为开放的OWASP Top 10挑战注册，并获得基于xUbuntu的LiveCD，它免费为您提供了在开始时所需要的一切工具。

为开放的OWASP Top 1挑战登录并注册https://www.hacking-lab.com/events/registerform.html?eventid=245&uk=

下载LiveCD http://media.hacking-lab.com/largefiles/livecd/

(i) http://www.nwlink.com/~donclark/hrd/strategy.html

OWASP新闻

Michael Coates

security101@lists.owasp.org

OWASP创建了一个新的邮件列表，以将重点放在把安全信息传递给那些新加入安全领域的人士。有一个关于一个安全话题的问题吗？想知道一个特定话题被推荐的最佳实例是哪一个吗？加入security101邮件列表，提出您的问题或者帮助他人解答吧！

通过以下链接加入：https://lists.owasp.org/mailman/listinfo/security101

每月安全突击讨论

OWASP开始在每月进行一次有关安全的突击讨论，我们将集合我们的安全社区围绕某一特定主题进行讨论。主题可能是一个漏洞、防御性的设计方法、技术，甚至一种方法论。安全社区的所有成员都鼓励编写有关本月话题的博客、文章、工具补丁、视频等。我们的目标是展示不同的开发人员、破坏者和维护人员的不同观点。

https://www.owasp.org/index.php/OWASP_Security_Blitz

OWASP确认的会员Linkedin组

想和其他OWASP会员进行社交活动吗？想让全世界都知道您支持OWASP吗？如果您是一名OWASP会员，加入OWASP确认的会员Linkedin组吧！注意：这是一张虚拟的徽章或会员卡。在Linkedin组中不提供其他资源或讨论。

http://www.linkedin.com/groups?viewMembers=&gid=4342746&sik=1336166179573

https://www.owasp.org/index.php/Membership

即将开展的活动

全球AppSec活动

全球AppSec活动	日期	地点	全球会议委员会代表	OWASP 开场介绍/关键陈述人
Global AppSec Research 2012 (Wiki)	2012年7月10日——2012年7月13日	希腊，雅典	John Wilander	Tom Brennan
Global AppSec North America 2012	2012年10月22日——2012年10月26日	美国，德克萨斯州，奥斯丁	Lorna Alamri	Michael Coats, Matt Tesauro, Tom Brennan, Eoin Keary
Global AppSec Latin America 2012	2012第四季	乌拉圭，蒙得维的亚	TBD	Tom Brennan
OWASP AppSec ASIAPAC 2013	2013年2月21日——22, 2013年2月22日	韩国，济州	TBD	TBD

地区和本地活动

AppSec India 2012	地区活动	2012年8月24日——2012年8月25日	印度	Tom Brennan
OWASP Ireland	地区活动	2012年9月4日——2012年9月6日	爱尔兰，都柏林	Eoin Kearny, Tom Brennan

合作伙伴和优惠活动

您期望在下表中出现您的活动吗？确保与全球会议委员会协作。

活动	日期	地点	OWASP出席代表
BHack Conference	2012年6月14日——2012年6月17日	巴西，Belo Horizonte/MG	TBD
Cyber Security, Cyber Warfare and Digital Forencis (CyberSec12)	2012年6月26日——2012年6月28日	吉隆坡	TBD
BlackHat USA	2012年7月25日——2012年7月26日	美国，拉斯维加斯	TBD

全球委员会

全球分部委员会

任务

为全球各地的分会提供所需的帮助，以使 OWASP 的总体任务和目标发展并对其作出贡献。

委员会主席：Josh Sokol

全球会议委员会

任务

OWASP 的全球会议委员会（GCC）的存在是为了协调全球的 OWASP 会议和活动，并为它们提供方便。

委员会主席：Mark Bristow

全球连接委员会

任务

帮助OWASP基金会一统一一致的方式对外进行沟通。我们还协助不同OWASP项目和委员会之间的内部沟通。

委员会主席：Jim Manico

全球教育委员会

任务

为公司企业、政府部门和教育机构提供对于应用安全的宣传、培训和教育服务。

委员会主席：Martin Knobloch

全球行业委员会

任务

OWASP全球行业委员会（GIC）旨在在行业、政府机构、学术界和监管机构中宣传和促进软件安全的最佳实例，并代表行业发出声音。这将通过各类宣传活动，包括：演讲、编写重要的文档，并与其他实体开展合作。

委员会主席：Rex Booth

全球会员委员会

任务

会员委员会推荐各种政策、流程和策略以加强OWASP成员的数值和质量。该委员会提供一个书面计划，并推荐政策、流程和措施，以确保OWASP是一个日益扩大的重要成员组织。

委员会主席：Dan Cornell

全球项目委员会

任务

营造一个积极的OWASP开发人员社区，为来自OWASP社区成员做出的贡献提供便利，为新项目提供支持和方向，并鼓励在大型国际组织采用OWASP的项目。

委员会主席：Jason Li

条款I - OWASP Bylaws

第1.01节. 办公室

基金会的主要办公室位于马里兰州的霍华德县。根据董事会的指派，或者根据基金会业务的需要，基金会可能在位于马里兰州以内或以外的其他地方设立办公室。

第1.02节. 目的

OWASP基金会将会是一个繁荣兴旺的全球社团，将会为全球软件业的安全带来革命。

第1.03节. 价值

开放：OWASP中的所有东西，从我们的财政状况到我们的代码，都是完全透明开放的。**创新：**OWASP鼓励并支持对于软件安全问题解决方案的创新和实验。**全球性：**全世界的任何人都被鼓励参加OWASP社团。**诚实：**OWASP是一个诚实的、不涉及厂商的全球社团。

业界合作伙伴


acunetix
ACCUVANT LABS
Adobe
ADP
CERT
Akamai
amazon
AppliedTrust
ascure
ASPECT SECURITY
Application Security Experts
ASTECH security
art defence
BEST BUY
Black Hat
Cargill
cigital
CORE SECURITY TECHNOLOGIES
DENIM GROUP
DIGITAL DEFENSE INCORPORATED
DREAMLAB TECHNOLOGIES
ENGINEERING
fishnet SECURITY
FUJITSU
gemalto
security to be free
GOTHAM DIGITAL SCIENCE
HARRIS CONNECT
HIGH-TECH BRIDGE
INFORMATION SECURITY SOLUTIONS
HUAWEI
IBM
(ISC)2
iMPERVA
Information Builders
IOActive
COMPREHENSIVE COMPUTER SECURITY SERVICES
mozilla Firefox
INTREPIDUS GROUP
mnemonic
securing your business
nixu
NetIQ
NOKIA
ORACLE
Penta SECURITY
PRAETORIAN
protiviti
Independent Risk Consulting
QUALYS
ON DEMAND SECURITY
Quotium
rackspace
Redspin
salesforce.com
Success On Demand
SD ELEMENTS
Security Innovation
SECURITY PS
Seeker
Symantec
TENABLE
Network Security
Trustwave
ups
WhiteHat
SECURITY

学术界合作伙伴


ADELPHI UNIVERSITY
Anglia Ruskin University
AUM
CIT
Cork Institute of Technology
DARTMOUTH
VOX CLAMANTIS IN DESERTO
1769
DAKOTA STATE
dsu
UNIVERSITY
DCU
foi
UNIVERSITY OF ZAGREB
FACULTY OF ORGANIZATION AND INFORMATICS
VARAŽDIN
KATHOLIEKE UNIVERSITEIT
LEUVEN
hochschule mannheim
heig-vd
Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud
HOCHSCHULE FURTWANGEN UNIVERSITY
HFU
IDC HERZLIYA
Efi Arazi School of Computer Science
ISCTE IUL
Instituto Universitário de Lisboa
Lisbon University Institute
TECNOLÓGICO DE MONTERREY
isep
itb
Institute of Technology Blanchardstown
Institiúid Teicneolaíochta Baile Bhlainséir
NYP
Nanyang Polytechnic
ORT
UNIVERSIDAD ORT
Uruguay
RUTGERS
THE STATE UNIVERSITY OF NEW JERSEY
TRINITY COLLEGE DUBLIN
SGU
STEVENS
INSTITUTE of TECHNOLOGY
TEMASEK POLYTECHNIC
UCIRVINE
NKU
NORTHERN KENTUCKY UNIVERSITY
northumbria UNIVERSITY
NORTH TEXAS
THE GEORGE WASHINGTON UNIVERSITY
WASHINGTON DC
NYU·poly
POLYTECHNIC INSTITUTE OF NYU
ΠΑΡΙΣΑΣ
FACULTAD DE INGENIERIA
Universidad de Buenos Aires
UCLA
UdeMM
Universidad de la Marina Mercante
UT DALLAS

OWASP基金会

开放的 Web 应用安全项目（OWASP）是一个国际的专业安全社团，致力于帮助企业和组织设计、开发、获取、操作和维护安全的应用系统。为了改善应用软件的安全，OWASP 的所有工具、文件、论坛和分会都是免费和开源的。我们认为应用安全的问题是人、流程和技术的问题，因为同时处理这三个问题是到达应用安全的最佳途径。OWASP 是一个新型的组织。由于没有商业压力，我们可以提供应用安全方面的的公正、实用和有效的信息。虽然 OWASP 提倡使用商业技术，但是我们与任何技术公司都没有关联。跟许多开源项目类似，OWASP 以合作和公开的方式制作了多种应用安全材料供大家使用。

核心价值

- 开放性——OWASP中的一切，从我们的财政状况到我们的代码，从根本上都是透明的；
- 创新性——OWASP鼓励并支持对于软件安全问题解决方案的创新和实验；
- 全球性——世界各地的人都被鼓励参加OWASP社团；
- 完整性——OWASP是一个诚实的、可信任的、厂商中立的和国际的社团。

OWASP是一个专业的应用程序安全开放社区。
参与这个组织的机会是无限制的

Donate

OWASP会员制

OWASP基金会专业联盟是一个不以营利为目的、不与任何商业业务或服务想联系的501c3慈善机构。要取得成功，我们需要您的支持。OWASP的个人，以及应用安全社团支持的教育和商业组织共同协作，创建了文章、方法、文档、工具和技术。
OWASP 所有成员的完整列表，可以在这里找到：https://www.owasp.org/index.php/Membership。

个人支持者 50美金/每年

- 强调您的 web 应用软件安全的意识；
- 享有参加 OWASP 会议的折扣；
- 扩大您的个人社交网；
- 获取一个 owasp.org 的电子邮件地址；
- 分配您会员费的 40%，以直接支持你选择的地方分会；
- 参与全球选举，并在社团的关键问题上拥有投票权。

企业支持者 5000美金/每年

- 减税捐赠；
- 享有在 OWASP 会议展示的产品或服务的折扣；
- 有机会在 owasp.org 的网站上免费粘贴 30 天的广告（价值$ 2,500）；
- 在 OWASP 的网站上粘贴您公司的徽标，以被确认为 OWASP 的支持者；
- 在每季度的新闻简报中被列为一个支持者，新闻简报将被发送给上万个个人；
- 通过全球行业委员会代表，有一个集体的声音；
- 参与全球选举，并在社团的关键问题上拥有投票权；
- 分配你每年捐赠的 40%，以直接支持你选择的地方分会和/或项目。

有关会员及赞助机会的更多信息，请与 Kelly Santalucia 联系： Kelly.santalucia@owasp.org

JOIN NOW

OWASP会员类型

	在选举期间拥有发言权	出现在OWASP.Org网站上	会议折扣	免费广告	出现在新闻简报中	owasp.org邮件地址	直接支持各地分会或项目
企业会员	X	X	X	X	X	X	X
个人会员	X	X	X		X	X	X
政府支持者		X	X		X		X
学术支持者		X	X		X	X	
组织支持者		X	X	X	X		X

其他支持OWASP的方式

地区分会支持者

尚未有兴趣成为一个完整企业会员的组织，但有更多愿望支持他们的地方分会，则可能更愿意成为一个本地分会支持者。请与您当地的分会领导核对，以了解更多有关分会支持者的具体经费等级。捐赠资金的 90％直接支持 OWASP 地方分会，10％分给 OWASP 基金会。地方分会网页

单一会议支持者

组织希望以 100%减税捐赠的形式支持 OWASP 地区分会，以使 OWASP 基金会能继续任务。具体费用由地方分会设定，如果您希望与地方分会合作，请与分会领导人联系。地方分会网页

活动赞助

通过赞助博览会或为与会人员提供赠品来参与我们的全球或区域活动。查看赞助机会

减税捐赠

OWASP 基金会是一个在美国和欧洲分别注册为 501（c）和不盈利的实体。因此，您的直接捐赠被认为是慈善捐赠。请联系您的税务顾问以了解全部信息。

个人参与

全球共有超过 140 个活跃的分部，超过百个 OWASP 项目，以及数以万计的想法正等待成为项目，这就非常容易找到一种方式加入进来。需要参与的是一个愿意分享想法，并与同行业中的核心专家合作。请联系您的地方分会领导人、某个项目领导者，或您自己开始吧！

新闻简报广告业务

- 1/4 单页广告：2000 美金；
- 1/2 单页广告：2500 美金；
- 1/2 单页广告 + 一个 OWASP 网站中的标题横幅或者 10 本 Top 10 的书：3000 美金；
- 一整页单页广告：5000 美金；
- 全年（每季度的新闻简报中占有 1/2 单页广告）：9000 美金。

欲知详情，请联系：Kelly.Santalucia@owasp.org或 Kate.Hartmann@owasp.org。