Heartbleed 攻击实验

1 概要

Heartbleed 漏洞（CVE-2014-0160）是 OpenSSL 库的一个严重实现缺陷，它允许攻击者从受害服务器的内存中窃取数据。被盗数据的内容取决于服务器内存中的内容。它可能包含私钥、TLS 会话密钥、用户名、密码、信用卡等信息。该漏洞存在于 Heartbeat 协议的实现中，该协议用于保持 SSL/TLS 活跃的连接状态。

本实验的目的是让学生了解 Heartbleed 漏洞的危害性、攻击的工作原理以及如何修复此问题。受影响的 OpenSSL 版本范围从 1.0.1 到 1.0.1f。SEEDUbuntu 12.04 虚拟机中的版本为 1.0.1。

参考资料和视频 有关 Heartbleed 攻击的详情可以在以下内容中找到：

- SEED 教材第 20 章，*Computer & Internet Security: A Hands-on Approach*, 3rd Edition, by Wenliang Du. 详情请见 `https://www.handsonsecurity.net`.
- SEED 视频第 11 节，*Internet Security: A Hands-on Approach*, by Wenliang Du. 详情请见 `https://www.handsonsecurity.net/video.html`.

实验环境 此实验已在我们预先构建的 Ubuntu 12.04 虚拟机上进行测试，可以从 SEED 网站下载此虚拟机。如果你正在使用我们的其它虚拟机，则此攻击不会成功，因为该漏洞已经被修复。你可以从 SEED 网站下载 SEEDUbuntu12.04 虚拟机。如果你有 Amazon EC2 帐户，可以从 Community AMIs 找到我们的虚拟机，其名称为 SEEDUbuntu12.04-Generic。需要注意的是，亚马逊的网站称这是一个 64 位虚拟机，但这是错的，因为该虚拟机是 32 位的。然而，此错误信息不会引起任何问题。

2 实验环境

在本实验中，我们需要设置两个虚拟机：一个称为攻击机器，另一个称为受害服务器。我们使用预先构建的 `SEEDUbuntu12.04` 虚拟机。这些虚拟机需要使用 `NAT-Network` 适配器进行网络设置。可以通过进入虚拟机设置、选择网络并点击适配器标签将适配器切换为 `NAT-Network` 来实现这一点。确保两个虚拟机都在同一个 NAT 网络上。

用于此攻击的网站可以是任何 HTTPS 网站，只要是使用 SSL/TLS。然而，由于攻击真实网站是违法的，我们已经在我们的虚拟机中设置了一个网站，并将在自己的虚拟机上进行攻击。我们使用一个开源的社交网络应用程序 `ELGG` ，此网站的 URL 为：`https://www.heartbleedlabelgg.com`。

我们需要修改攻击机器上的 `/etc/hosts` 文件，将服务器名称映射到托管 `ELGG` 应用程序的服务器虚拟机的 IP 地址。在 `/etc/hosts` 中搜索以下行，并将 IP 地址 `127.0.0.1` 更改为实际的服务器虚拟机的 IP 地址，该虚拟机托管了 `ELGG` 应用程序。

```
127.0.0.1   www.heartbleedlabelgg.com
```

3 实验任务

在进行实验任务之前，你需要理解 Heartbeat 协议是如何工作的。Heartbeat 协议由两种消息类型组成：HeartbeatRequest 包和 HeartbeatResponse 包。客户端向服务器发送一个 HeartbeatRequest 包。当服务器接收到它时，会在 HeartbeatResponse 包中发送一个副本以保持连接活跃。

3.1 任务 1：发起 Heartbleed 攻击

在这个任务中，学生将对我们的社交网络站点发起 Heartbleed 攻击，并观察可以造成什么样的损害。Heartbleed 攻击的实际损害取决于服务器内存中存储的信息类型。如果服务器上活动不多，则无法窃取有用的资料。因此，我们需要以合法用户的身份与网页服务器交互。让我们以管理员身份操作并执行以下步骤：

- 使用浏览器访问 `https://www.heartbleedlabelgg.com`
- 以站点管理员身份登录。(用户名：`admin`；密码：`seedelgg`)
- 将 Boby 添加为好友。(前往 `More -> Members` 并点击 `Boby -> Add Friend`)
- 向 Boby 发送私人消息

在你已经用合法用户和系统交互之后，你可以发起攻击并查看可以从受害服务器获取什么信息。从头开始编写 Heartbleed 攻击程序并不容易，因为它需要掌握 Heartbeat 协议的底层知识。幸运的是，其他人已经写好了攻击代码。因此我们将使用现有的代码来体验 Heartbleed 攻击。我们使用的代码称为 `attack.py`，最初是由 Jared Stafford 写的。为了教育目的，我们在代码中做了一些小改动。你可以从实验网站下载代码，更改其权限以使文件可执行。然后可以按照以下方式运行攻击代码：

```
$ ./attack.py  www.heartbleedlabelgg.com
```

你可能需要多次运行攻击代码才能获取有用的信息。尝试从中获取以下信息。

- 用户名和密码
- 用户的活动（用户做了什么）
- 私人消息的具体内容

对于从 Heartbleed 攻击中窃取的每一项秘密，你需要展示屏幕截图作为证据，并解释你是如何发起攻击的，以及你的观察结果。

3.2 任务 2：查找 Heartbleed 漏洞的原因

在这个任务中，学生将比较由攻击代码发送的正常消息包和恶意消息包的结果，以找出 Heartbleed 漏洞的根本原因。

Heartbleed 攻击基于 Heartbeat 请求。此请求只是向服务器发送一些数据，服务器会复制这些数据到其响应包中，所以所有数据都会被回显回去。在正常情况下，假设请求包括 3 个字节的数据 ABC，因此长度字段的值为 3。服务器将在内存中放置数据，并从数据的开头开始复制 3 个字节到其响应包中。在攻击场景中，请求可能包含 3 个字节的数据，但长度字段可能为 1003。当服务器构建其响应包时，它将从数据的起始位置 (即 ABC) 复制数据，但实际上复制的是 1003 个字节而不是 3 个字节。这些额外的 1000 字节显然不是来自请求包中的，它们来自服务器的内存，可能包含其他用户的资料、秘密密钥、密码等。

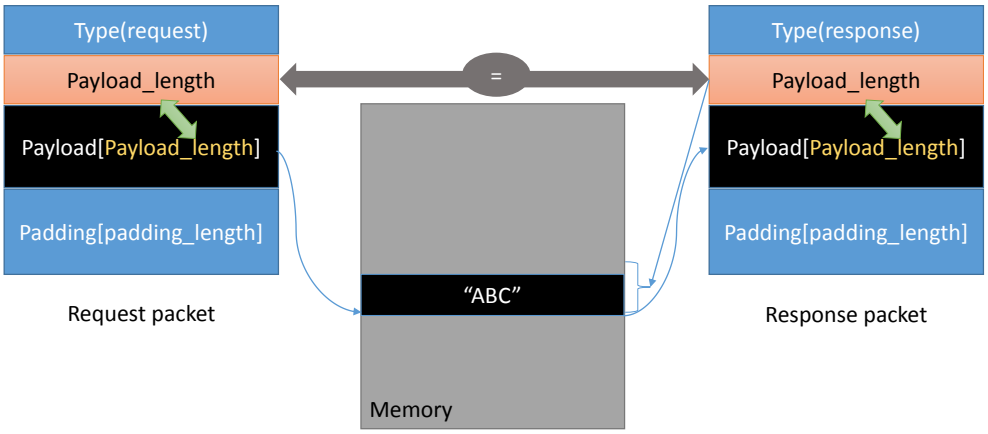

图 1: 正常的 Heartbeat 通信

在这个任务中，我们将修改请求中的长度字段。首先，请从图 1 了解如何构建 Heartbeat 响应包。当收到 Heartbeat 请求包时，服务器将从该包获取有效载荷和 `Payload_length` 字段的值（在图 1 中突出显示）。这里，有效载荷只是一个 3 字节字符串 `ABC` 而且 `Payload_length` 的值正好是 3。服务器程序将从请求包中获取此长度值。然后从存储 ABC 的内存复制 `Payload_length` 个字节数来构建响应包。这样，响应包就会包含一个 3 字节字符串 ABC。

我们可以通过如图 2 所示的方式发起 HeartBleed 攻击。我们保持相同的有效载荷（3 字节），但将 `Payload_length` 字段设置为 1003。当服务器构建响应包时，它会无条件地从请求包中获取此 `Payload_length` 值，然后从存储字符串 ABC 的内存复制 1003 个字节到响应包作为有效载荷。除了字符串 ABC 外，额外的 1000 字节也被复制到响应包中，这些字节是内存中的其它内容，例如私密活动、日志信息、密码等。

我们的攻击代码允许你使用不同的 `Payload_length` 值进行操作。默认值是一个相当大的数值 (`0x4000`)，但你可以通过命令选项 `"-l"` (小写字母 l) 或 `"--length"` 来减少大小，如下所示：

```
$./attack.py  www.heartbleedlabelgg.com  -l  0x015B
$./attack.py  www.heartbleedlabelgg.com    --length 83
```

你的任务是使用不同大小的有效载荷长度值运行攻击程序并回答以下问题：

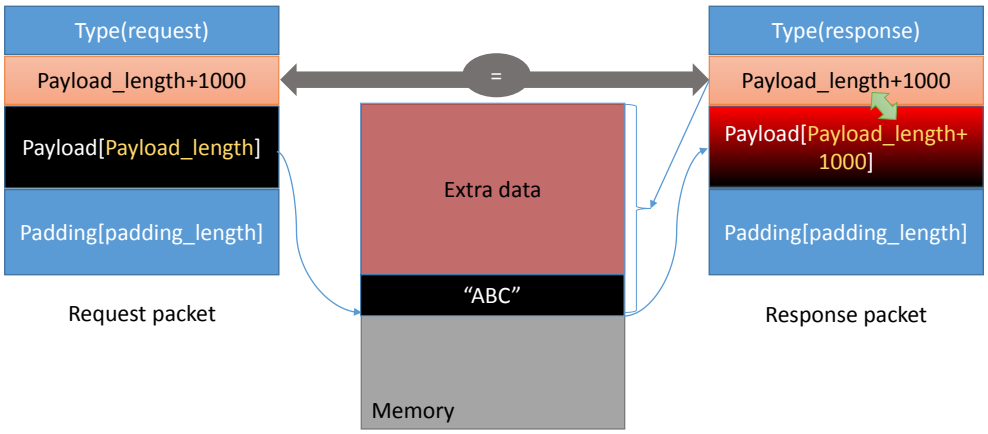

图 2: Heartbleed 攻击

- **问题 2.1：** 随着长度变量的减小，你能观察到什么差异？
- **问题 2.2：** 当长度变量减小时，有一个输入长度变量的边界值。在低于该边界时，Heartbeat 查询将收到响应包而不附加任何额外的数据（这意味着请求是正常的）。请找到这个边界长度。你可能需要尝试许多不同的长度值，直到 web 服务器发送回没有额外数据的回复。为了帮助你完成此任务，在返回字节数小于预期长度时，程序会打印 `"Server processed malformed Heartbeat, but did not return any extra data."`

3.3 任务 3：缓解措施与补丁修复

要修复 Heartbleed 漏洞的最好方法是将 OpenSSL 库更新到最新版本。这可以通过以下命令实现。需要注意的是，一旦更新后很难回到易受攻击的版本。因此，在进行更新之前，请确保完成先前的任务。你也可以在更新前为你的虚拟机做一个 snapshot 备份。

```
$ sudo apt-get update
$ sudo apt-get upgrade
```

任务 3.1 在更新 OpenSSL 库之后再次尝试你的攻击。请描述你的观察结果。

任务 3.2 本任务的目标是找出如何在源代码中修复 Heartbleed 漏洞。以下 C 语言结构（不完全是源代码）是 Heartbeat 请求/响应包的格式。

```
struct {
   HeartbeatMessageType type; // 1 字节: 请求或响应
   uint16 payload_length;     // 2 字节: 有效载荷长度
   opaque payload[HeartbeatMessage.payload_length];
   opaque padding[padding_length];
} HeartbeatMessage;
```

包的第一个字段(1 字节)是类型信息,第二个字段(2 字节)是有效载荷长度,接着是实际的 payload 和填充。有效载荷的大小应该与 payload_length 字段中的值相同,但在攻击场景中,payload_length 可以设置为不同的值。以下代码片段显示了服务器如何从请求包复制数据到响应包。

Listing 1: 处理 Heartbeat 请求包并生成响应包

```
/* 分配一个用于响应的内存，大小是 1 字节消息类型加上 2 字节有效载荷长度，再加上有效载荷和填充 */

unsigned int payload;
unsigned int padding = 16; /* 使用最小填充 */

// 首先从 type 字段读取
hbtype = *p++;   /* 在此指令执行后，指针 p 将指向 payload_length 字段。*/

// 从请求包中的 payload_length 字段读取
n2s(p, payload); /* 函数 n2s(p, payload) 从指针 p 中读取 16 位并将其存储在 INT 变量 "payload" 中。*/

// pl 指向有效载荷内容的起始位置

if (hbtype == TLS1_HB_REQUEST)
{
     unsigned char *buffer, *bp;
     int r;

     /* 分配一个用于响应的内存，大小是 1 字节消息类型加上 2 字节有效载荷长度，再加上有效载荷和填充 */

     buffer = OPENSSL_malloc(1 + 2 + payload + padding);
     bp = buffer;

     // 放入响应类型和长度，并复制有效载荷
     *bp++ = TLS1_HB_RESPONSE;
     s2n(payload, bp);

     // 复制有效载荷
     memcpy(bp, pl, payload); /* pl 是指向有效载荷内容起始位置的指针 */

     bp += payload;

     // 随机填充
     RAND_pseudo_bytes(bp, padding);

     // 从缓冲区复制 3+payload+padding 字节到 Heartbeat 响应包中，并发送给请求客户端。
     r = ssl3_write_bytes(s, TLS1_RT_HEARTBEAT, buffer,
           3 + payload + padding);
     OPENSSL_free(buffer);
}
```

请指出 Listing 1 中代码的问题，并提供修复该问题的解决方案（即需要进行哪些修改来修复此问题）。你不需要重新编译代码；只需在你的实验报告中描述如何修复此问题。

此外，请评论 Alice、Bob 和 Eva 对 Heartbleed 漏洞根本原因的不同讨论：Alice 认为根本原因是缺少缓冲区复制时的边界检查；Bob 认为原因是缺少用户输入验证；Eva 认为我们可以通过删除包中的长度值来解决一切。

4 提交

你需要提交一份带有截图的详细实验报告来描述你所做的工作和你观察到的现象。你还需要对一些有趣或令人惊讶的观察结果进行解释。请同时列出重要的代码段并附上解释。只是简单地附上代码不加以解释不会获得学分。