Spoofing Location
Cet entête est utilisé pour demander une redirection. Pouvant être modifié et redirigé sur un site pirate, son exploitation présente un risque.

Spoofing GET & POST
Comme les autres informations, les données transmises via un GET et/ou unPOST peuvent également être modifiées avant leur envoi au serveur.

Exploitation par injection de code
Les entêtes étant modifiables, ils peuvent également servir à des attaques par injection de codes dont voici un exemple.
Supposons que l'on remplace :

Referer=http://www.serge-ungar.com/Vulnérabilités.php

Par

Referer=<?php fopen(backdoor.php); ?>

En fonction du code de l'application et du serveur, il est possible que «backdoor.php» soit exécuté avec les conséquences que l'on peut imaginer.

Attention, en fonction des paramètres de Header modifiés, l'attaque peut être faite contre le serveur lui-même et non contre le site. Par exemple l'établissement de statistiques est réalisé par des programmes «serveurs» qui exploitent les informations des «referer»

Sécurisation
D'une façon générale, il est préférable de passer par le tableau de variables adressé par $_SERVER qui donne accès à toutes les variables des entêtes mais également aux méthodes. Ce tableau fournit également de nombreuses autres informations comme $_SERVER['REMOTE_ADDR'] qui retourne l'adresse IP de l'internaute en train de visualiser la page dans laquelle cette commande est insérée. Si l'internaute est derrière un proxy, on obtient l'adresse IP du proxy.

Les données récupérées à travers ces variables doivent toujours être filtrées afin de minimiser les risques. Les filtres dépendent bien évidement des applications qui utilisent les valeurs de ces variables.

Vulnérabilité CRLF
Présentation
Que ce soit sous Windows ou sous Linux, il existe des caractères spécifiques pour annoncer une fin de ligne. Sous Windows on utilise la combinaison de caractères spéciaux \r (Carriage Return qui vaut 10 en ASCII et %0A en héxadécimal) suivi de \n (Line Feed qui vaut 13 en ASCII et %0D en héxadécimal); sous Unix on n'utilise que \n. Ce sont exactement les mêmes caractères utilisés pour tous les protocoles Internet de la couche Applications pour annoncer une fin de commande ou de données.

Détection

Dans le cas présent, la détection de la vulnérabilité se fait par son exploitation.

Exploitation

L'exploitation de cette vulnérabilité consiste à faire suivre ces caractères d'un code malicieux qui peut être une redirection de site ou des commandes systèmes. Les deux cas les plus classiques d'exploitation de cette vulnérabilité sont les mails et les headers mais cette vulnérabilité est applicable dans de nombreux autres cas (login, forum, formulaires, …)

Application de la vulnérabilité aux mails

Considérons un site permettant d'envoyer un mail à l'administrateur en précisant l'expéditeur. On va avoir un code du genre :

```
$To = "webmaster@site.com"
$Sujet = "Test";
$From = "moi@mail.com";
$Texte = "Bonjour";
mail($To,$Sujet,$From,$Texte);
```

Le pirate va essayer de profiter de cette fonctionnalité pour envoyer un mail anonyme à plusieurs personnes. Cette attaque ne peut se faire dans le champ «Texte» car le premier saut de ligne est assimilé à un morceau du corps du message. Il n'en est rien concernant le champ $From dans lequel on va pouvoir insérer des headers supplémentaires. Supposons que le champ $Form soit rempli par :

$From = "pirate@pirate.com%0ACc:dest1@dest.com,dest2@dest2.com%0ABcc:dest3@dest3.com"

Le mail se présentera sous la forme :

```
$To = "webmaster@site.com"
To = webmaster@site;com
Subject = Test
From = pirate@pirate.com
Cc= dest1@dest1.com,dest2@dest2.com
Bcc=dest3@dest3.com
Text=Bonjour
```

Et le pirate aura la possibilité d'envoyer des mails à d'autres destinataires en utilisant le site Internet de la victime.

Application de la vulnérabilité aux headers

Le protocole HTTP utilise les caractères spéciaux \n\r comme fin de ligne et notamment pour séparer les différents headers. Considérons les headers suivants qui vont être envoyés par le navigateur au serveur :

- Host=www.site.com
- User-Agent=Mozilla/5.0 (Windows; U; Windows NT 6.0; fr; rv:1.8.1.11) Gecko/20071127 Firefox/2.0.0.11
- Accept=text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png,*/*;q=0.5
- Accept-Language=fr,fr-fr;q=0.8,en-us;q=0.5,en;q=0.3
- Accept-Encoding=gzip,deflate
- Accept-Charset=ISO-8859-1,utf-8;q=0.7,*;q=0.7
- Keep-Alive=300

- Connection=keep-alive
- Referer=http://www.site.com/index.php
- Cache-Control=max-age=0

Le pirate peut modifier cet header pour essayer de renvoyer le serveur sur un autre site qui lui permettra d'injecter un code malicieux. Pour ce faire, il peut modifier l'entête Connection en la remplaçant par :

Connection=keep-alive\n\rLocation=www.pirate.com/index.php\n\r

Le reste des headers sera alors ignoré.

Sécurisation

Les données envoyées par mail doivent être filtrées pour supprimer des headers les caractères spéciaux \n,\r mais aussi %0A, %0D, 0x0A et 0x0D.

Vulnérabilités XSS

Présentation

Certainement la plus connue et la plus populaire des vulnérabilités. XSS signifie Cross Site Scripting. Elle consiste à faire exécuter du code Javascript, HTML, … sur le navigateur de la victime.

De quoi vient-elle ? Tout simplement du fait que les données transmises d'une page à l'autre sont à un moment ou un autre affichées à travers une fonction du type «echo($var)» pour le PHP. S'il n'y a pas d'affichage, il n'y a évidemment pas de vulnérabilité car c'est le navigateur qui interprète ce que lui a envoyé la fonction «echo».

On peut également exécuter du code VBScript. Il suffit de préciser dans la balise, le langage à utiliser, ce qui donne:

- <script language="vbscript">msgbox "Vulnérabilité XSS ouverte"; </script>

Avec du javascript, du vbscript, …, on peut récupérer des informations comme les cookies, les logins, les mots de passe, et de nombreuses autres données qui sont stockées dans le navigateur de la victime.

On distingue quatre types de vulnérabilités XSS :

- Les vulnérabilités non-permanentes ou par réflexion
- Les vulnérabilités permanentes ou par stockage
- Les vulnérabilités par injection DOM
- Les scripts cachés dans des iframes

La particularité de cette vulnérabilité est qu'elle nécessite une action généralement non comprise ou non vue de l'internaute. Le pirate ne peut agir seul.

Détection

Considérons le code HTML suivant qui permet l'accès à un compte:

```
<form name="form1" method="post" action="login.php">
<tr>
        <td>Login</td>
```