HTTPS

目录
通过 HTTPS 进行管理
强制 HTTPS 登录与 HTTPS 管理访问 {#to-force-ssl-logins-and-ssl-admin-access}
示例 {#example}
使用反向代理 {#using-a-reverse-proxy}
Example {#example-2}
注意
更多信息 {#further-information}
Virtual Hosts {#virtual-hosts}
摘要 {#summary}
验证 {#verification}
Limitations {#limitations}

HTTPS 是一种加密通信协议 ——
本质上是一种更安全的网页浏览方式，因为它能在您的浏览器和网络服务器之间建立专用通道。这就是为什么
大多数主流网站都采用 HTTPS。

如果网站使用 HTTPS，您会在地址栏看到一个小锁图标，如下图所示：

“安全网站”锁形图标截图

以下是在您自己的网站上使用 HTTPS 的最常见原因：

更快。 有人可能认为 HTTPS 会让网站变慢，因为加密和解密所有数据需要时间。但许多 HTTP
的效率改进只有在使用 HTTPS 时才能实现。因此，对于绝大多数访问者来说， HTTPS
实际上会让您的网站加载更快。

信任。
用户更容易信任安全的网站。虽然他们不一定知道自己的流量已被加密，但他们确实明白小锁图标意味着网站
重视隐私。技术人员会知道，您的计算机与网络服务器之间的任何中间服务器都无法查看或篡改传输的信息。

支付安全。 如果您在网站上销售商品，用户希望确保支付信息安全。 HTTPS
和小锁图标能保证他们的信息安全传输到网络服务器。

搜索引擎优化。 许多搜索引擎会对不使用 HTTPS 的网站进行降权处理，从而使其更难在搜索结果中获得靠前排名。

您的声誉。 您是否注意到某些网站地址旁会显示 “不安全 ”字样？

当您的网络浏览器想让您知道某个网站未使用 HTTPS 时就会出现这种情况。浏览器希望您明白（这种想法是正确的！）：那些懒得使用 HTTPS （很多情况下是免费的）的网站所有者不值得您花费时间，更不值得您花钱。

反过来，您肯定也不希望浏览器暗示您可能是那种不可靠的网站所有者。

当 TLS/SSL 证书已安装并可供网络服务器使用时， WordPress 完全兼容 HTTPS。强烈建议启用 HTTPS 支持，以帮助维护 WordPress 登录和网站访问者的安全性。

通过 HTTPS 进行管理

要轻松启用（并强制）通过 SSL 进行 WordPress 管理，您可以在网站的 wp-config.php 文件中定义两个常量。在插件文件中定义这些常量是不够的；必须在您的 wp-config.php 文件中定义。您还必须已在服务器上配置好 SSL ，并为安全服务器配置好（虚拟）主机，然后您的网站在将这些常量设置为 true 时才能正常工作。

注意： `FORCE_SSL_LOGIN` 已在 4.0 版本中弃用。请使用 `FORCE_SSL_ADMIN` 。

强制 HTTPS 登录与 HTTPS 管理访问 {#to-force-ssl-logins-and-ssl-admin-access}

在 `wp-config.php` 文件中将常量 `FORCE_SSL_ADMIN` 设置为 true，即可强制所有登录 及 所有管理会话通过 SSL 进行。

示例 {#example}

```
define( 'FORCE_SSL_ADMIN', true );
```

使用反向代理 {#using-a-reverse-proxy}

如果 WordPress 托管在提供 SSL 的反向代理之后，但其自身未启用 SSL ，这些设置最初会导致请求陷入无限重定向循环。为避免此问题，你可以配置 WordPress 以识别 `HTTP_X_FORWARDED_PROTO` 标头（假设你已正确配置反向代理来设置该标头）。

Example {#example-2}

```
define( 'FORCE_SSL_ADMIN', true );
// in some setups HTTP_X_FORWARDED_PROTO might contain
// a comma-separated list e.g. http,https
// so check for https existence
if( strpos( $_SERVER['HTTP_X_FORWARDED_PROTO'], 'https') !== false )
$_SERVER['HTTPS'] = 'on';
```

注意

当您使用代理转发重定向时，您将请求传输到网络中的某个主机，但不会传输与之关联的标头。然而，
WordPress 需要某些标头才能执行一些重定向。为了传输这些标头，您需要在重定向配置中添加一些行。

例如，对于 Nginx，您需要添加以下行：

```
location / {
proxy_pass http://your_host_name:your_port;
proxy_set_header Host $host:$server_port;
proxy_set_header X-Real-IP $remote_addr;
proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
proxy_set_header X-Forwarded-Host $server_name;
proxy_set_header X-Forwarded-Proto $scheme;
proxy_redirect off;
}
```

像 `$variable` 这样的变量由反向代理自动管理。

更多信息 {#further-information}

本文其余部分提供的信息适用于以下情况：您正在使用旧版 WordPress（理想情况下不应如此！）或您的 SSL 设置有所不同（例如，您的 SSL 证书用于其他域名）。

有时，您希望整个 wp-admin 通过 https 协议在安全连接上运行。从概念上讲，该过程的工作原理如下：

1. 设置两个具有相同 URL（博客 URL）的虚拟主机，一个安全，另一个不安全。
2. 在安全的虚拟主机上，设置重写规则，将所有非 wp-admin 流量转移到不安全站点。
3. 在不安全的虚拟主机上，设置重写规则，将所有 wp-admin 流量转移到安全主机。
4. 通过插件添加过滤器，过滤 wp-admin 中的链接，以便激活后，管理链接被重写为使用 https，并编辑 cookie 使其仅通过加密连接工作。

□□□□□□□ WordPress 1.5 □□□ `mod_rewrite` □ Apache□□ `httpd.conf` □□□□□□□□□□□ `.htaccess` □□□□□□□□□□□□□□□□□□□□□□

Virtual Hosts {#virtual-hosts}

You need a (virtual) host configured for the secure server in addition to the non-secure site. In this example, the secure virtual host uses the same `DocumentRoot` as the insecure host. Hypothetically, you could use a host with a different name, such as wpadmin.mysite.com and link the document root to the wpadmin directory.

Please ask your ISP to set up a secure virtual host for you, or if you have administrative access set up your own. Note that you cannot use name based virtual hosting to identify different SSL servers.

Rewrite Rules For The Insecure Host

In the `.htaccess` or virtual host stanza in `httpd.conf` for your insecure host, add this rewrite rule to automatically go to the secure host when you browse to https://example.com/wp-admin/ or https://example.com/wp-login.php

This should go above the main wordpress rewrite block.

```
RewriteCond %{THE_REQUEST} ^[A-Z]{3,9}\ /(.*)\ HTTP/ [NC]
RewriteCond %{HTTPS} !=on [NC]
RewriteRule ^/?(wp-admin/|wp-login\.php) https://example.com%{REQUEST_URI}%{QUERY_STRING}
[R=301,QSA,L]
```

If you are using permalink rewrite rules, this line must come before `RewriteRule ^.*$ - [S=40]`.

An important idea in this block is using `THE_REQUEST`, which ensures only actual http requests are rewritten and not local direct file requests, like an include or fopen.

Rewrite Rules For Secure Host (Optional)

These rewrite rules are optional. They disable access to the public site over a secure connection. If you wish to remain logged in to the public portion of your site using the plugin below, you must *not* add these rules, as the plugin disables the cookie over unencrypted connections.

The secure virtual host should have two rewrite rules in an .htaccess file or in the virtual host declaration (see Using Permalinks for more on rewriting):

```
RewriteRule !^/wp-admin/(.*) - [C]
RewriteRule ^/(.*) https://www.example.com/$1 [QSA,L]
```

The first rule excludes the wp-admin directory from the next rule, which shuffles traffic to the secure site over to the insecure site, to keep things nice and seamless for your audience.

Setting WordPress URI

For some plugins to work, and for other reasons, you may wish to set your WordPress URI in options to reflect the https protocol by making this setting https://example.com. Your blog address should not change.

Example Config Stanzas

NOTE: The below config is not 100% compatible with WordPress 2.8+, WordPress 2.8 uses some files from the wp-includes folder. The redirection that the first set of Rewrite rules introduces may cause security warnings for some users. See #10079 for more information.

```
<VirtualHost nnn.nnn.nnn.nnn:443>
□ServerName www.example.com

SSLEngine On
□SSLCertificateFile /etc/apache2/ssl/thissite.crt
□SSLCertificateKeyFile /etc/apache2/ssl/thissite.pem
□SetEnvIf User-Agent ".*MSIE.*" nokeepalive ssl-unclean-shutdown

□DocumentRoot /var/www/mysite

□<IfModule mod_rewrite.c>
□□RewriteEngine On
□□RewriteRule !^/wp-(admin|includes)/(.*) - [C]
□□RewriteRule ^/(.*) https://www.example.com/$1 [QSA,L]
□</IfModule>

</VirtualHost>
```

□□□□□

```
<VirtualHost *>
□ServerName www.mysite.com

□DocumentRoot /var/www/ii/mysite

□<Directory /var/www/ii/mysite >
□□<IfModule mod_rewrite.c>
□□□RewriteEngine On
□□□RewriteBase /
```

```
□□□RewriteCond %{REQUEST_FILENAME} -f [OR]
□□□RewriteCond %{REQUEST_FILENAME} -d
□□□RewriteRule ^wp-admin/(.*) https://www.example.com/wp-admin/$1 [C]
□□□RewriteRule ^.*$ - [S=40]
□□□RewriteRule ^feed/(feed|rdf|rss|rss2|atom)/?$ /index.php?&feed=$1 [QSA,L]

□□</IfModule>
□</Directory>

</VirtualHost>
```

□□□□□□□□□□

□□□□□□□□□□ SSL □□□□□□□□□□□□□□□□ RewriteRules□

□□□

```
RewriteRule ^/wp-(admin|login|register)(.*) https://www.example.com/wp-$1$2 [C]
```

□□

```
RewriteRule !^/wp-(admin|login|register)(.*) - [C]
```

□□□ **443** □□□ **80** □□□□□□□□□□□

```
# BEGIN WordPress
<IfModule mod_rewrite.c>
RewriteEngine On
RewriteBase /

# □□□□□ 443 □□□□□□□□ SSL □ HTTP□□□□□□□
RewriteCond %{SERVER_PORT} !^80$
RewriteRule !^wp-(admin|login|register)(.*) - [C]
RewriteRule ^(.*)$ https://%{SERVER_NAME}/$1 [L]

# □□□□□ 80□HTTP□□□□□□□
RewriteCond %{SERVER_PORT}  ^80$
RewriteCond %{REQUEST_FILENAME} -f [OR]
RewriteCond %{REQUEST_FILENAME} -d
RewriteRule ^wp-(admin|login|register)(.*) https://%{SERVER_NAME}:10001/wp-$1$2 [L]
```

```
RewriteCond %{SERVER_PORT}  ^80$
RewriteCond %{REQUEST_FILENAME} !-f
RewriteCond %{REQUEST_FILENAME} !-d
RewriteRule . /index.php [L]

</IfModule>
```

摘要 {#summary}

此方法 无法 修复 WordPress 中的某些 固有安全风险_____
，也不能保护您免受中间人攻击或其他可能破坏安全连接的风险。

然而，它 应该 能大大增加恶意人员窃取您的 Cookie 和 /或身份验证标头，并利用它们冒充您访问 wp-admin 的难度。它还能混淆嗅探您内容的能力，这对于可能包含需要严格保护的文档草稿的法律博客来说可能很重要
。

验证 {#verification}

在作者的服务器上，日志显示 GET 和 POST 请求均通过 SSL 进行，并且所有发往不安全主机上 wp-admin 的流量都被转移到了安全主机。

示例 POST 日志行:

```
[Thu Apr 28 09:34:33 2005] [info] Subsequent (No.5) HTTPS request received for child 6 (server foo.com:443)
xx.xxx.xxx.xxx - - [28/Apr/2005:09:34:33 -0500] "POST /wp-admin/post.php HTTP/1.1" 302 - "https://foo.com/wp-admin/post.php?action=edit&post=71" "Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.7.7) Gecko/20050414 Firefox/1.0.3"
```

进行更多测试（最好使用数据包嗅探器和一些专业的网络分析工具）将有助于进一步确认。

Limitations {#limitations}

The author assumes (but hasn't checked) that if the user has stored cookies/told their browser to remember passwords (not based on form fields but if using certain external auth mechanism) and hits https://www.example.com/wp-admin/, those packets are sent in the clear and the cookie/auth headers could be intercepted. Therefore, to ensure maximum security, the user should explicitly use the https host or always log in at the beginning of new sessions.

版本号 #2
由 壳壳虫 创建于 2026-03-26 17:08:37 UTC
由 壳壳虫 更新于 2026-03-26 17:08:37 UTC