作为最流行的 Web 服务器，Apache Server 提供了较好的安全特性，使其能够应对可能的安全威胁和信息泄漏。

1、 采用选择性访问控制和强制性访问控制的安全策略

从 Apache 或 Web 的角度来讲，选择性访问控制 DAC（Discretionary Access Control）仍是基于用户名和密码的，强制性访问控制 MAC（Mandatory Access Control）则是依据发出请求的客户端的 IP 地址或所在的域号来进行界定的。对于 DAC 方式，如输入错误，那么用户还有机会更正，从新输入正确的的密码；如果用户通过不了 MAC 关卡，那么用户将被禁止做进一步的操作，除非服务器作出安全策略调整，否则用户的任何努力都将无济于事。

２、Apache 的安全模块

Apache 的一个优势便是其灵活的模块结构，其设计思想也是围绕模块（Modules）概念而展开的。安全模块是 Apache Server 中的极其重要的组成部分。这些安全模块负责提供 Apache Server 的访问控制和认证、授权等一系列至关重要的安全服务。

mod_access 模块能够根据访问者的 IP 地址(或域名，主机名等)来控制对 Apache 服务器的访问，称之为基于主机的访问控制。

mod_auth 模块用来控制用户和组的认证授权（Authentication）。用户名和口令存于纯文本文件中。mod_auth_db 和 mod_auth_dbm 模块则分别将用户信息（如名称、组属和口令等）存于 Berkeley-DB 及 DBM 型的小型数据库中，便于管理及提高应用效率。

mod_auth_digest 模块则采用 MD5 数字签名的方式来进行用户的认证，但它相应的需要客户端的支持。

mod_auth_anon 模块的功能和 mod_auth 的功能类似，只是它允许匿名登录，将用户输入的 E-mail 地址作为口令。

SSL（Secure Socket Lager），被 Apache 所支持的安全套接字层协议，提供 Internet 上安全交易服务，如电子商务中的一项安全措施。通过对通讯字节流的加密来防止敏感信息的泄漏。但是，Apache 的这种支持是建立在对 Apache 的 API 扩展来实现的，相当于一个外部模块，通过与第三方程序的结合提供安全的网上交易支持。

Apache 具有灵活的设置，所有 Apache 的安全特性都要经过周密的设计与规划，进行认真地配置才能够实现。Apache 服务器的安全配置包括很多层面，有运行环境、认证与授权设置等。Apache 的安装配置和运行示例如下：

1、以 Nobody 用户运行

一般情况下，Apache 是由 Root 来安装和运行的。如果 Apache Server 进程具有 Root 用户特权，那么它将给系统的安全构成很大的威胁，应确保 Apache Server 进程以最可能低的权限用户来运行。通过修改 httpd.conf 文件中的下列选项，以 Nobody 用户运行 Apache 达到相对安全的目的。

User nobody

Group

-1

2、ServerRoot 目录的权限

为了确保所有的配置是适当的和安全的，需要严格控制 Apache 主目录的访问权限，使非超级用户不能修改该目录中的内容。Apache 的主目录对应于 Apache Server 配置文件 httpd.conf 的 Server Root 控制项中，应为：

Server Root /usr/local/apache

3、SSI 的配置

在配置文件 access.conf 或 httpd.conf 中的确 Options 指令处加入 Includes NO EXEC 选项，用以禁用 Apache Server 中的执行功能。避免用户直接执行 Apache 服务器中的执行程序，而造成服务器系统的公开化。

Options Includes Noexec

4、阻止用户修改系统设置

在 Apache 服务器的配置文件中进行以下的设置，阻止用户建立、修改 .htaccess 文件，防止用户超越能定义的系统安全特性。

AllowOveride None

Options None

Allow from all

然后再分别对特定的目录进行适当的配置。

5、改变 Apache 服务器的确省访问特性

Apache 的默认设置只能保障一定程度的安全，如果服务器能够通过正常的映射规则找到文件，那么客户端便会获取该文件，如 http://local host/~ root/ 将允许用户访问整个文件系统。在服务器文件中加入如下内容：

order deny,ellow

Deny from all

将禁止对文件系统的缺省访问。

6、CGI 脚本的安全考虑

CGI 脚本是一系列可以通过 Web 服务器来运行的程序。为了保证系统的安全性，应确保 CGI 的作者是可信的。对 CGI 而言，最好将其限制在一个特定的目录下，如 cgi-bin 之下，便于管理；另外应该保证 CGI 目录下的文件是不可写的，避免一些欺骗性的程序驻留或混迹其中；如果能够给用户提供一个安全性良好的 CGI 程序的模块作为参考，也许会减少许多不必要的麻烦和安全隐患；除去 CGI 目录下的所有非业务应用的脚本，以防异常的信息泄漏。

以上这些常用的举措可以给 Apache Server 一个基本的安全运行环境，显然在具体实施上还要做进一步的细化分解，制定出符合实际应用的安全配置方案。

Apache Server 默认情况下的安全配置是拒绝一切访问。假定 Apache Server 内容存放在/usr/local/apache/share 目录下，下面的指令将实现这种设置：

Deny from all

Allow Override None

则禁止在任一目录下改变认证和访问控制方法。

同样，可以用特有的命令 Deny、Allow 指定某些用户可以访问，哪些用户不能访问，提供一定的灵活性。当 Deny、Allow 一起用时，用命令 Order 决定 Deny 和 Allow 合用的顺序，如下所示：

1、 拒绝某类地址的用户对服务器的访问权（Deny）

如：Deny from all

Deny from test.cnn.com

Deny from 204.168.190.13

Deny from 10.10.10.0/255.255.0.0

2、 允许某类地址的用户对服务器的访问权（Allow）

如：Allow from all

Allow from test.cnn.com

Allow from 204.168.190.13

Allow from 10.10.10.0/255.255.0.0

Deny 和 Allow 指令后可以输入多个变量。

3、简单配置实例：

order Allow, Deny

Allow from all

Deny from www.test.com

指想让所有的人访问 Apache 服务器，但不希望来自 www.test.com 的任何访问。

order Deny, Allow

Deny from all

Allow from test.cnn.com

指不想让所有人访问，但希望给 test.cnn.com 网站的来访。

概括的讲，用户认证就是验证用户的身份的真实性，如用户帐号是否在数据库中，及用户帐号所对应的密码是否正确；用户授权表示检验有效用户是否被许可访问特定的资源。在 Apache 中，几乎所有的安全模块实际上兼顾这两个方面。从安全的角度来看，用户的认证和授权相当于选择性访问控制。

建立用户的认证授权需要三个步骤：

1、建立用户库

用户名和口令列表需要存在于文件（mod_auth 模块）或数据库（mod_auth_dbm 模块）中。基于安全的原因，该文件不能存放在文挡的根目录下。如，存放在 /usr/local/etc/httpd 下的 users 文件，其格式与 UNIX 口令文件格式相似，但口令是以加密的形式存放的。应用程序 htpasswd 可以用来添加或更改程序：

htpasswd –c /usr/local/etc/httpd/users martin

-c 表明添加新用户，martin 为新添加的用户名，在程序执行过程中，两次输入口令回答。用户名和口令添加到 users 文件中。产生的用户文件有如下的形式：

martin:WrU808BHQai36

jane:iABCQFQs40E8M

art:FadHN3W753sSU

第一域是用户名，第二个域是用户密码。

2、配置服务器的保护域

为了使 Apache 服务器能够利用用户文件中的用户名和口令信息，需要设置保护域（Realm）。一个域实际上是站点的一部分（如一个目录、文档等）或整个站点只供部分用户访问。在相关目录下的.htaccess 文件或 httpd.conf（acces.conf）中的< Directory>段中，由 AuthName 来指定被保护层的域。在.htaccess 文件中对用户文件有效用户的授权访问及指定域保护有如下指定：

AuthName “restricted stuff”

Authtype Basic

AuthUserFile /usr/local/etc/httpd/users

Require valid-user

其中，AuthName 指出了保护域的域名（Realm Name）。valid-user 参数意味着

user 文件中的所有用户都是可用的。一旦用户输入了一个有效的用户/口令时，同一个域内的其他资源都可以利用同样的用户/口令来进行访问，同样可以使两个不同的区域共用同样的用户/口令。

3、告诉服务器哪些用户拥有资源的访问权限

如果想将一资源的访问权限授予一组客户，可以将他们的名字都列在 Require 之后。最好的办法是利用组（group）文件。组的操作和标准的 UNIX 的组的概念类似，任一个用户可以属于一个和数个组。这样就可以在配置文件中利用 Require 对组赋予某些权限。如：

Require group staff

Require group staff admin

Require user adminuser

指定了一个组、几个组或一个用户的访问权限。

需要指出的是，当需要建立大批用户帐号时，那么 Apache 服务器利用用户文件数据库将会极大地降低效率。这种情况下，最好采用数据库格式的帐号文件，譬如 DBM 数据库格式的文件。还可以根据需要利用 db 格式（mod_auth_db）的数据文件，或者直接利用数据库，如：mSQL （mod_auth_msql）或 DBI 兼容的数据库（mod_auth_dbi）。

DBM 文件是一种简单而标准的用于加快读取效率的保存信息的方法。文件中存

放的每一个记录由两个部分组成部分：键和值。由于 DBM 的格式，使得与键相关的信息非常有效。在 Web 用户认证中，这里的键将是用户名，而与该键相关的值将是该用户经过加密的口令信息。从 DBM 文件中查找用户名和口令，要比从一个纯文本文件中查找有效得多。对于有很多用户的站点，这种方法将大大提高用户认证的效率。

1、在 Apache 服务器中增加 DBM 模块

在默认的条件下，Apache 不使用 DBM 文件来完成用户认证，因此编译时一定要加入可选的 DBM 认证模块。重新配置 Apache 服务器文件，去掉其中的注释行

Module dbm_auth_module mod_auth_dbm.o 前的“

”，并重新编译。但是，在编译之前，需要指出 Apache DBM 函数的位置。

2、创建 DBM 用户文件（假设文件名为 users）

Apache 提供了一个“dbmmanage”的程序，用于创建和管理 DBM 文件。其中：

Dbmmanage /usr/local/etc/httpd/usersdbm 创建 DBM 文件

Dbmmanage /usr/local/etc/httpd/users adduser martin hamster 新增用户

Dbmmanage /usr/local/etc/httpd/usersdbm delete martin 删除用户

Dbmmanage /usr/local/etc/httpd/usersdbm view 显示 DBM 中所有用户

有了 DBM 数据库文件，还要替换目录访问控制，即将 Apache 配置文件（access.conf）中的 AuthUserFile 部分替换成： AuthUserFile /usr/local/etc/usersdbm 告诉 Apache 现在的用户文件是 DBM 的格式。

由于 Apache 服务器本身并没有内置 ASP 功能，因此我们需要第三方软件的合适的 ASP 模块来支持；目前 Apache 支持 ASP 的产品（模块）有这么几种：SUN ONE ASP、iASP、Apache::ASP、OpenASP、Mod_gb 和 ModVB 等；本文就以 SUN 公司的 SUN ONE ASP 4.0 产品为例，来讲解如何让 ASP 运行在 Apache 服务器上，操作系统的环境采用 windows 2003 server。

Sun Java System Active Server Pages 4.0（以前称为 Sun ONE Active Server Pages）是一个安全的跨平台 Active Server Pages (ASP) 引擎。Java System ASP 允许组织通过将部署的 ASP 应用程序从 Microsoft IIS 提升到在 Solaris 操作系统、Linux 或 Windows 上运行的 Java System Web Server（以前称为 Sun ONE Web Server）或 Apache，提高 Web 安全性。Java System ASP 与 Microsoft ASP 3.0 和诸如 Macromedia 的 Dreamweaver MX 、Microsoft FrontPage 等通用的 Web 撰写工具完全兼容。

Sun ONE Active Server Pages 4.0 允许在多种 Web 服务和平台上部署用 Active Server Pages 编写的 Web 应用。这就意味着 Microsoft Web 开发人员可以使用现有工具创建跨平台 Active Server Pages 应用，而机构可以在安全、高度可

用的 Web 服务器和操作环境中部署基于 Active Server Pages 软件的内容。

说明：只有安装好 JRE（SUN ONE ASP 软件本身就已附带了，不需要另外再下载 J2SE 软件）的环境和 Apache 服务器软件后，才能正式安装 SUN ONE ASP 4.0.2 版本的软件。

在 SUN ONE ASP 软件安装好之后，我们重起 Apache Web Server；在下图 32 和 33 中，我们看到 SUN ONE ASP 整合到 Apache 中 httpd.conf 文件具体模块说明（是自动装载的）。这样就可以让 Apache 支持 asp 了.

Linux+Apache 的稳定性、安全性和性能以及低廉的价格正在赢得越来越多的市场份额，使用 Linux+Apache 作网站服务器的朋友也越来越多，而 Apache 作为一种 http 服务，相比 FTP 总是不容易控制，特别是当网站以 http 方式提供软件/音乐下载时，若是每个用户都开启多个线程并没有带宽的限制，将很快达到 http 的最大连接数或者造成网络壅塞，使得网站的许多正常服务都无法运行。不过，Apache 的使用者们早已开发出了 mod_limitipconn 和 mod_bandwidth 两个模块，来控制 http 的并发连接数和用户所能够使用的带宽，下面将以 RedHat Linux 7.3+Apache 1.3.7 来说明它们的使用方法。

一、使用 mod_limitipconn 限制 Apache 的并发连接数

mod_limitipconn 可以控制每个 IP 地址同时连接服务器某一个目录的并发连接数，

是一个非常有用的模块,其官方网页是http://dominia.org/djao/limitipconn.html，最新版本为 for Apache 1.3.7 的 0.04，并且还有支持 Apache 2.x 的模块下载，由于本人使用 Apache 1.3.7 版本，所以请使用 2.x 版本 Apache 的朋友到其官方网站察看具体的使用方法。

mod_limitipconn for Apache 1.3x 提供三种安装方式，分别是 tar 包、rpm 安装文件和 rpm 源文件，由于 rpm 包只能用在 RedHat 7.x 版本，并且不支持检测代理服务器，所以我们一般都使用 tar 包的安装方式。

以管理员方式登陆服务器，然后在服务器上运行 wget http://dominia.org/djao/limit/mod_limitipconn-0.04.tar.gz 将 mod_limitipconn 的 tar 包下载到服务器，然后按照再运行 tar zxvf mod_limitipconn-0.04.tar.gz 将 tar 包解压缩，会在当前目录下生成 mod_limitipconn-0.04 目录，然后 cd mod_limitipconn-0.04 进入此目录，下一步就是使用 apxs 将目录中的 mod_limitipconn.c 编译。这时，我们需要确定自己的 Apache 安装在那个目录，并且找到 apxs 命令放在哪里。

通过命令 whereis apxs，我们可以确定 apxs 命令的路径，如我的 apxs 命令所在为 /usr/sbin/apxs，则输入 /usr/sbin/apxs -c -i -a mod_limitipconn.c 对 mod_limitipconn.c 进行编译，此命令会自动在你 Apache 的配置文件 httpd.conf 中加入需要的信息，并且将生成的 mod_limitipconn.so 模块拷贝到 Apache 的模块目录。不过为了确认此命令是否正常运作，请首先检查自己的 Apache 模块目

录（我的是/usr/lib/apache），看内部是否含有 mod_limitipconn.so 文件，没有的话请将 mod_limitipconn-0.04 目录中生成的文件拷贝到此处。

刚才命令自动生成的 httpd.conf 可能有些错误，在我的系统中，它将 LoadModule limitipconn_module modules/mod_limitipconn.so 放在了

LoadModule python_module modules/mod_python.so

之间，而将 AddModule mod_limitipconn.c 放在了

AddModule mod_python.c

之间，直接造成了 mod_limitipconn 模块不能正常运行，所以请将这两行分别移动到没有< /IfDefine>的相应行中，然后请确认 mod_status 模块已经加载，并且在 mod_status 下添加了 ExtendedStatus On 这一行。这时我们的 mod_limitipconn 模块就安装完毕，下一步就是对某个目录进行并发连接数的设置了。

mod_limitipconn 可以对全局和虚拟主机进行不同的限制，其语法结构都是

所限制的目录所在，此处表示主机的根目录

MaxConnPerIP 3

所限制的每个 IP 并发连接数为 3 个

NoIPLimit image/*

对图片不做 IP 限制

所限制的目录所在，此处表示主机的/mp3 目录

MaxConnPerIP 1

所限制的每个 IP 并发连接数为 1 个

OnlyIPLimit audio/mpeg video

该限制只对视频和音频格式的文件

当对全局进行限制时，将这段代码放在 httpd.conf 文件没有 VirtualHost 的地方，若是对某个虚拟主机进行限制，请将其放在< VirtualHost xxx.xxx.xxx.xxx>和之间，我们可以通过更改 Location 以及 MaxConnPerIP 方便的控制所限制的目录和并发连接数。

最后，只要重新启动 Apache 服务，并发连接数的限制就可以生效。

二、使用 mod_bandwidth 控制 Apache 的带宽

Apache 1.3.7 实际上带有 mod_bandwidth 支持，只是没有此模块的 so 文件，我们所做的就是下载 mod_bandwidth 的源文件进行编译，并对 mod_bandwidth 进行相应的设置。

在下载之前，请先确认自己的 Apache 配置文件 httpd.conf 中是否含有

LoadModule bandwidth_module modules/mod_bandwidth.so

以及

AddModule mod_bandwidth.c

若是没有，请加上

LoadModule bandwidth_module

libexec/apache/mod_bandwidth.so

AddModule mod_bandwidth.c

并且这两行必须分别加在相应区域的最前面，使得这个模块以最低的优先级运行。（不过 1.3.7 的 Apache 应该有，呵呵）。

确认后，请输入 wget ftp://ftp.cohprog.com/pub/apache/module/1.3.0/mod_bandwidth.c 将源文件下载到服务器，然后请使用 apxs 对其进行编译，编译方法和 mod_limitipconn 的基本相同，如我输入 /usr/sbin/apxs -c mod_bandwidth.c -o /usr/lib/apache(Apache 的模块目录)，编译程序会自动将编译成功的 mod_bandwidth.so 文件放到 Apache 的模块目录，您也可以自己确认一下，若是不正常，拷贝过去即可。

mod_bandwidth 运行时需要一些特定的目录，按照默认情况，请运行以下命令创建并更改目录的权限：

```
mkdir /tmp/apachebw
mkdir /tmp/apachebw/link
mkdir /tmp/apachebw/master
chmod -R 777 /tmp/apachebw
```

然后再打开 httpd.conf 文件，加上以下内容

BandWidthDataDir "/tmp/apachebw/"

BandWidthModule on

这时，我们就能够对所需要限制带宽的目录进行相应的设置，此处的目录请使用服务器的绝对路径。如我们想限制服务器/home/www/thinkjam/download/soft 目录的下载速度，也就是限制网址 http://download.thinkjam.org/soft 目录下软件的下载速度，则为 httpd.conf 文件增加以下内容

BandWidth thinkjam.org 0

来自 thinkjam.org 的下载不受速度限制

BandWidth 210.51.21 0

来自 210.51.21 网段的下载不受速度限制

BandWidth all 327680

来自其它网段的速度都限制为 327680Byte，即 30KB/s

设置完毕后，重新启动 Apache 服务，即可生效。

mod_bandwidth 还有许多其它有用的参数，如在中间加上 MaxConnection 120 则可以限制某个目录的最多连接数，当超过指定连接数时，拒绝新的连接，此参

数与 mod_limitipconn 模块结合可以控制某个目录的最多连接人数。

其它的参数请朋友们到其官方网站 http://www.cohprog.com/v3/bandwidth/doc-en.html 察看相关的文档。

Apache 的功能确实强大，很多功能都可以通过添加模块来实现，在 http://modules.apache.org/ 可以找到更多的模块，我们也可以编写自己的模块来实现相应的功能。 apache 服务器实现用户验证 apache 服务器已经内置用户验证机制，大家只要适当的加以设置，便可以控制网站的某些部分要用户验证。大家只要跟着我一步步做下来就应该能轻松实现用户验证。 第 1 步： 我们在/var/www(apache 的主页根目录)下建立一个 test 目录 mkdir /var/www/test 第 2 步 然后我们编辑 httpd.conf 添加 Alias /test"/var/www/test" Options Indexes MultiViews AllowOverride AuthConfig 表示进行身份验证 order allow,deny Allow from all AllowOverride AuthConfig 表示进行身份验证 这是关键的设置 第 3 步 在 /var/www/test 创建 .htaccess 文件 vi /var/www/test/.htaccess AuthName "frank share web" AuthType Basic AuthUserFile /var/www/test/.htpasswd require valid-user AuthName 描述，随便写 AuthUserFile /var/www/test/.htpasswd require valid-user 或者 require user frank 限制是所有合法用户还是指定用户 密码文件推荐使用.htpasswd,因为 apache 默认系统对“.ht”开头的文件默认不允许外部读取，安全系数会高一点哦。 第 4 步 就是创建 apache 的验证用户 htpasswd -c /var/www/test/.htpasswd frank 第一次创建用户要用到-c 参数 第 2 次添加用户，就不用-c 参数 如果你们想修

改密码，可以如下 htpasswd -m .htpasswd frank 第 5 步： ok，重启 apache 服务，然后访问 http://你的网站地址/test 如果顺利的话，应该能看到一个用户验证的弹出窗口，只要填入第 4 步创建的用户名和密码就行 后话，为了服务器的性能，一般不推荐使用 AllowOverride AuthConfig 或者 AllowOverride ALL，因为这会使服务器会不断的去寻找.htaccess,从而影响服务器的效能，一般我们把一些后台管理界面或者其他特殊目录可能需要加验证这个需求。