WWW サーバの構築

川本芳久
大阪大学基礎工学部
kawamoto@es.osaka-u.ac.jp

1 WWW サーバ (Apache)

WWW サーバソフトはいろいろありますが，最近よく使われるようになった Apache について，プログラムのコンパイルや設定方法について説明します．

Apache は，Apache HTTP Server Project によって開発されている UNIX で動く WWW サーバで，フリーソフトウェアです．以前よく使われていた NCSA httpd バージョン 1.3 が元になっていますので，NCSA httpd を運用していたところではすぐにでも移行が可能です．

2 インストール

まず，最新版の Apache のソースファイルを手に入れます．現在の Apache の最新バージョンは 1.1.1(apache_1.1.1.tar.gz) ですが，頻繁にバージョンアップがあるので作成する時点での最新版を手に入れるようにします．

2.1 コンパイル

まず，Apache Version 1.1.1 のファイル apache_1.1.1.tar.gz を展開するディレクトリを決めます．これをサーバルートと呼びますが，ここでは/var/www/apache_1.1.1 に展開すると決めたとして説明します．

始めに，ディレクトリ/var/www で apache_1.1.1.tar.gz を展開します．

```
$ cd /var/www
$ gzip -cd < ~/apache_1.1.1.tar.gz | tar xf -
```

これでファイルが apache_1.1.1 の下に展開されます．

コンパイルするために変更が必要なファイルは，src/Configuration です．Apache でサポートされているシステム (表 1) 以外では src/conf.h の変更も必要です．

表 1: Apache でサポートされているシステム

SunOS 4.x	Digital UNIX (DEC OSF/1)	SCO OpenServer 5	FreeBSD
SunOS 5.x(Solaris 2.x)	NeXT	SVR4	BSD/OS 2.x
SGI IRIX	Sequent	UnixWare 2.x	QNX
HP-UX	Linux	Amdahl UTS 2.1	LynxOS
AIX	A/UX	HP/Apollo Domain/OS	DG/UX 5.4
Ultrix	SCO ODT 3	NetBSD	EMX OS/2

またApacheでは，追加機能がモジュールとして用意されています．モジュールの設定もsrc/Configurationで行います．モジュールの機能についての説明がsrc/README に書いてありますので，これを参照しながらどのモジュールを追加すればよいか決めてください．

- src/Configuration

  AUX_CFLAGS,AUX_LIBS 各システム用の定義が書いてあるので，インストールするシステムにあった定義の部分を有効にします．

  Module さまざまなモジュール機能をサーバに組み込みます．

- src/conf.h

  /* Unknown system - Edit these to match */以下の行をシステムにあわせて変更します．
  各 #define の意味は src/README ファイルに説明があります．

変更ができれば make を以下のように実行します．

```
% cd src
% ./Configure
% make
```

以上の操作で Apache のコンパイルが終わります．また，ファイルのインストールという作業は特に必要ありませんが，実行ファイルは /usr/sbin/ ディレクトリなどにコピーしておいた方がよいでしょう．

```
% su
Password:
# cp src/httpd /usr/sbin/.
```

3 設定

3.1 設定ファイル

Apache の動作は，httpd.conf, srm.conf, access.conf というの 3 つの設定ファイルにより設定されます．通常の設定では，conf/ ディレクトリに設定ファイルを用意します．なお，conf/ディレクトリには設定ファイルの元になる httpd.conf-dist, srm.conf-dist, access.conf-dist というファイルが用意されていますので，これらをコピーして使用するとよいでしょう．

これらの設定ファイルの書き方ですが，記述形式はすべて同じです．

- アルファベットの大文字と小文字の区別はない．

  ただしUNIX のファイル名は大文字と小文字を区別しますので，ファイル名の部分のみは例外になります．

- コメント行は#で始まる．

  行の 1 文字目が#だと，その行はコメントとして扱われます．行の途中から #でコメントを書くことはできません．

- 1 行には 1 つの命令 (directive)．

  1 行には 1 つだけ命令を書くことができます．1 行の文字数の制限はありませんが，複数の行にまたがる記述はできません．

- 余分な空白は無視される.

  命令のデータの区切りにはスペースかタブを使いますが, 余分な空白は無視されます. また, データ中に空白を入れたい時には, 空白の前に \ が必要です.

3.2 httpd.conf

httpd.conf は Apache が起動時に最初に読み込む設定ファイルで, 主に UNIX のプロセスとして動作するための設定を行ないます. このファイルは Apache の実行時にオプションとして指定します.

ServerType Apache を inetd デーモンから実行するか, それともデーモンとして実行するかを決定します. 値には inetd あるいは standalone のどちらかを指定します. この命令を省略した場合, standalone が指定されているものとみなされます.

通常は inetd を指定します.

例

```
ServerType standalone
```

Port Apache が使用するポート番号を指定します. Port 命令を省略した場合, HTTP の標準ポート番号である 80 が使われます. また, この命令は ServerType が standalone の場合のみ有効です.

例

```
Port 80
```

HostnameLookups ファイルへ WWW クライアントからのアクセスを記録する時に, マシン名で記録するか, IP アドレスで記録するかを指定します. on の場合はマシン名で, off の場合は IP アドレスで記録されます.

例

```
HostnameLookups on
```

User 指定されたユーザの権限でファイルの読み込みやプログラムの実行を行ないます. ユーザ名あるいは#に続けてユーザ ID を指定します. User 命令を省略した場合, ユーザ ID −1 が使用されます.

例

```
User nobody
```

Group 指定されたグループの権限でファイルの読み込みやプログラムの実行を行ないます. グループ名あるいは#に続けてグループ ID を指定します. Group 命令を省略した場合, グループ ID −1 が使用されます.

例

```
Group #-1
```

ServerAdmin WWW サーバの管理者の e-mail アドレスを指定します. これは WWW サーバが, クライアントからのリクエストに対してエラーを返す時に使用されます. この命令を省略した場合, エラーには e-mail アドレスが含まれません.

例

```
ServerAdmin www-admin@es.osaka-u.ac.jp
```

ServerRoot サーバルートとなるディレクトリを指定します.

サーバ設定ファイルにおいてパス名を指定する命令では, "/" から始まっているパス名を除き, サーバルートからの相対パス名だとみなされます.

例

```
ServerRoot /var/www
```

ServerName WWW サーバのマシン名を指定します. gethostbyname 関数で得られるホスト名がドメイン名を含まない時や別名を持っている時などに指定します.

例

```
ServerName www.es.osaka-u.ac.jp
```

TimeOut 何かのトラブルで WWW クライアントとの通信ができなくなった場合のために, サーバとクライアントの間の通信の最大待ち時間を指定します. 単位は秒です. この命令を省略した場合, この時間は 1200 秒になります.

例

```
TimeOut 300
```

ErrorLog WWW サーバで起こったエラーを記録するファイル名を指定します. この命令を省略した場合, logs/error_log にエラーが記録されます.

例

○ サーバルートの下の logs/error_log に記録する場合

```
ErrorLog logs/error_log
```

○ /var/adm/httpd_errors に記録する場合

```
ErrorLog /var/adm/httpd_errors
```

○ エラーを記録しない場合

```
ErrorLog /dev/null
```

TransferLog WWW クライアントからのアクセスを記録するファイル名を指定します. この命令を省略した場合, logs/access_log にアクセスが記録されます.

例

```
TransferLog logs/access_log
```

PidFile httpd のプロセス ID を記録するファイル名を指定します. この命令を省略した場合, logs/httpd.pid に記録されます. またこの命令は, ServerType が standalone である場合のみ有効です.

例

```
PidFile /var/run/httpd.pid
```

AccessConfig アクセス設定ファイルを指定します. この命令を省略した場合, Apache は conf/access.conf をアクセス設定ファイルとして読み込みます.

例

```
AccessConfig conf/access.conf
```

ResourceConfig リソース設定ファイルのファイル名を指定します. この命令を省略した場合, conf/srm.conf が使用されます.

例

```
ResourceConfig conf/srm.conf
```

IdentityCheck TransferLog 命令で指定されるアクセス記録ファイルにリモートユーザ名を記録するかどうかを On または Off で指定します. リモートユーザとは WWW クライアントを使用しているユーザのことですが, リモートユーザ名は常に得られるとは限りません.

この命令を省略した場合, アクセス記録にはリモートユーザ名は記録されません.

例

```
IdentityCheck On
```

3.3 srm.conf

srm.conf ファイルでは, Apache が動作するのに必要なファイルについての情報を設定します. srm.conf 自体はサーバ設定ファイルの ResourceConfig 命令で指定しますが, 指定がなければサーバルートの下の conf/srm.conf というファイルが使用されます.

DocumentRoot Apache で公開するディレクトリをドキュメントルートと呼びますが, これを指定します. この命令を省略した場合, サーバルートの下の htdocs が使用されます.

例

```
DocumentRoot /var/www/htdocs
```

UserDir WWW サーバにアカウントのあるユーザは各自のホームディレクトリの下に個人のページを持つことができます. その個人のページを置くディレクトリ名を指定するのが UserDir 命令です. この命令を省略した場合, public_html/ディレクトリが使われます. また, キーワード DISABLED を指定した場合, この機能は使用できなくなります.

例

○ /~ユーザ名/ をアクセスすると ~ユーザ名/public_html/を参照するように設定する場合

```
UserDir public_html
```

○ 個人のページを使用させない場合

```
UserDir DISABLED
```

DirectoryIndex WWW サーバはディレクトリへのアクセスに対して, そのディレクトリ中に DirectoryIndex で指定されたファイルが存在するならば, そのファイルの内容をクライアントに返します. この命令を省略した場合, index.html というファイルが使われます.

例

```
DirectoryIndex index.html
```

AccessFileName ディレクトリごとのアクセス設定ファイル指定します. この命令を省略した場合, .htaccess というファイル名が使われます.

例

```
AccessFileName .htaccess
```

AddType ファイルの拡張子に対して, ファイルのタイプを新しく定義できます. 一般的なものはすでに定義済みですが, それらの定義を変更することもできます.

書式

```
AddType type/subtype extension
```

type/subtype は MIME タイプで, extension はファイルの拡張子, ファイル名を書くことができます.

例

○ .htm 拡張子を持つファイルは text/html と定義

```
AddType text/html htm
```

AddEncoding ファイルの MIME エンコーディングを拡張子によって定義することができます. 現在のところ, MIME エンコーディングには x-compress あるいは x-gzip が指定できますが, あまり定義する必要はないでしょう.

例

```
AddEncoding x-compress Z
AddEncoding x-gzip gz
```

DefaultType AddType で定義されなかったファイルに対して, MIME タイプをどう扱うかを指定します. この命令を省略した場合, DefaultType は text/plain(テキストドキュメント) になります.

例

○ MIME タイプの定義されていないファイルはバイナリファイルだと指定する場合

```
DefaultType application/octet-stream
```

Redirect アクセスの要求を他のサーバに転送します. 転送元のパス名と転送先の URL を指定します.

例

○ /images へのアクセスを http://www.osaka-u.ac.jp/images へ転送する場合

```
Redirect /images http://www.osaka-u.ac.jp/images
```

Alias ドキュメントルート以外のファイルを仮想的なパス名で参照させることができます. 仮想的なドキュメントのパス名と実際のファイルのパス名 (絶対パス) を指定します.

例

○ /var/ftp/pub を /ftp でアクセスできるようにする場合

```
Alias /ftp /var/ftp/pub
```

ScriptAlias CGI プログラムのパス名を設定します. パス名と実際に CGI プログラムがあるディレクトリ名を指定します.

また, ScriptAlias 命令のディレクトリ名の最後には/をつけておきます.

例

○ CGI プログラムが /var/www/cgi-bin/の下にある場合

```
ScriptAlias /cgi-bin/ /var/www/cgi-bin/
```

FancyIndexing WWW クライアントからのディレクトリへのアクセスの際に DirectoryIndex で指定されたファイルがない場合, Apache はディレクトリの中のファイル一覧を返すことができます. そのファイル一覧で表示される内容を変更することができます.

例

```
FancyIndexing on
```

ReadmeName ファイル一覧の最後には, ReadmeName で指定したファイルの内容を追加することができます. 指定されたファイル名.html というファイルをまず探し, 次にそのままの名前のファイルを探します.

例

○ README.html あるいは README の内容をファイル一覧の最後に追加する場合

```
ReadmeName README
```

HeaderName ファイル一覧の最初に, HeaderName で指定したファイルの内容を挿入します. この命令を省略した場合, ファイル一覧の最初には「Index of /ディレクトリ/」と表示されます.

例

○ HEADER.html あるいは HEADER の内容をファイル一覧の最後に追加する場合

```
HeaderName HEADER
```

AddDescription ファイル一覧のファイル名の後ろにつける説明を指定します. この命令を省略した場合, 説明はつきません.

書式

```
AddDescription "comment text" FileId
```

FileId には, 拡張子・パス名・ワイルドカードによるパターンのいずれかを指定することができます. ワイルドカードでは, *が任意の長さの文字列を, ?が任意の一文字を表しています.

説明文の "comment text"には空白を含めることができますが, 1 行に収める必要があります.

例

```
AddDescription "movie file" .mpeg
AddDescription "temporary file" test.html
AddDescription "html listing" /var/www/htdocs/list
AddDescription "photo image" *.photo.gif
```

AddIcon, AddIconByType, AddIconByEncoding ファイル一覧のアイコンとファイル名, MIME タイプ, MIME エンコーディングの対応関係を指定します. この命令を省略した場合, 対応関係は定義されません.

書式

```
AddIcon icon name1 name2...
AddIconByType icon name1 name2...
AddIconByEncoding icon name1 name2...
```

icon にはアイコンファイルのパス名を指定します. また, イメージを表示できない WWW クライアントがイメージのかわりに「[alt]」という説明を表示できるように, (alt,icon) という形式でも指定することができます.

name には, 拡張子, パス名, ワイルドカードによるパターンのいずれかを指定できます. さらに AddIcon の場合には^^DIREC-TORY^^, ^^BLANKICON^^を指定することができます. ^^DIRECTORY^^ はサブディレクトリのためのアイコンを, ^^BLANKICON^^は空白のためのアイコンを表します.

例

```
AddIcon /icons/image.xbm .gif .jpg .xbm
AddIconByType (SND,/icons/image.xbm) audio/*
AddIconByEncoding (CMP,/icons/compress.xbm) x-compress
```

DefaultIcon AddIcon などで対応関係が定義されなかったファイルに関して, アイコンの仮想パス名を指定します. この命令を省略した場合, アイコンは定義されません.

例

```
DefaultIcon /icons/unknown.xbm
```

IndexIgnore ファイル一覧に表示したくないファイル名を指定します. また, ファイル名の末尾や, ワイルドカードによるパターンを指定することもできます.

例

○ README ファイル, 名前が#や~で終了するファイル, 名前が . で始まるファイルを一覧に表示させない場合

```
IndexIgnore README # ~ /.??*
```

3.4 access.conf

Apache では, access.conf によってサーバへのアクセスに対し様々な制限を加えることができます. パスワードによるアクセスの制限, 情報の公開の制限, WWW サーバの機能の制限などです. access.conf は Apache 全体のアクセス制御を記述しますが, ディレクトリごとにアクセス制御を記述する.htaccess というファイルもあります.

⟨Directory⟩ アクセス制御命令の有効範囲を指定します. .htaccess ファイルでは ⟨Directory⟩ 命令は必要ありません.

例

○ /var/www/htdocs に関するアクセス制御を記述する例

```
<Directory /var/www/htdocs>
Options ...
AllowOverride ...
</Directory>
```

アクセス制御命令は, ⟨Directory⟩ 命令で指定したディレクトリだけでなく, それ以下のディレクトリに対しても有効です.

Options WWW サーバのどの機能を有効にするかを以下のオプションで記述します. この命令を省略した場合, MultiViews 以外のすべての機能が有効になります.

- **FollowSymLinks** シンボリックリンクの先を見るようにします.
- **SymLinksIfOwnerMatch** シンボリックリンクの先のファイルまたはディレクトリの所有者と, シンボリックリンクファイルの所有者とが同じ場合にのみ, シンボリックリンクの先を見るようにします.
- **ExecCGI** そのディレクトリにある CGI プログラムの実行を許可します.
- **Includes** そのディレクトリにある HTML 文書の server-side includes 機能を有効にします.
- **IncludesNoExec** exec 以外の server-side includes 機能を有効にします.
- **Indexes** ディレクトリのファイル一覧を Apache が作れるようにします. このオプションがないと Apache はファイル一覧を作りません.
- **MultiViews** 同じ URL に対して複数の言語を割り当てる機能を有効にします.
- **All** MultiViews 以外のオプションすべての機能を有効にします.
- **None** オプションすべての機能を無効にします.

例

```
Options SymLinksIfOwnerMatch IncludesNoExec Indexes
```

AllowOverride ディレクトリアクセス設定ファイルで有効にすることができる命令を, 次のオプションで指定します. この命令を省略した場合, すべての命令を有効にできます. この命令はグローバルアクセス設定ファイルでのみ使用できます.

- **Options** Options が有効になります.

- **FileInfo** AddType と AddEncoding が有効になります.
- **AuthConfig** AuthName, AuthType, AuthUserFile, AuthGroupFile が有効になります.
- **Limit** Limit が有効になります.
- **All** オプションのすべての命令が有効になります.
- **None** ディレクトリアクセス設定ファイルは無視されます.

例

```
AllowOverride Limit FileInfo
```

AuthName WWW サーバでは, ユーザ名とパスワードを要求することなどによって利用権の制限・確認を行なうことができます. このユーザ名・パスワードは, WWW サーバ使用に対してのみ使われるものであり, そのマシンに実在のユーザ名・パスワードとは関係ないものです. 利用権の制御は, 利用権に名前をつけてその名前ごとに行ないます. AuthName では, そのディレクトリとそれ以下のディレクトリで使用する利用権名を指定します. この命令を省略した場合, 利用権名は設定されません.

例

○ es_visitors という利用権名にする場合

```
AuthName es_visitors
```

AuthType 利用権の確認方法を指定しますが, Apache では Basic のみが指定できます. この命令を省略した場合, 利用権の確認方法は設定されません.

例

```
AuthType Basic
```

AuthUserFile 利用権の制御に使用する, ユーザ名とパスワードのリストが収められているパスワードファイル名を指定する命令です. この命令を省略した場合, パスワードファイルは設定されません.

このファイルは htpasswd という専用のコマンドで作成されるもので, UNIX システムのパスワードファイルとは異なります. システムのパスワードファイルを指定しないように注意して下さい.

例

```
AuthUserFile /var/www/conf/htpasswd
```

AuthGroupFile ユーザグループを定義したファイルの名前を指定します. この命令を省略した場合, グループファイルは設定されません.

例

```
AuthGroupFile /var/www/conf/htgroup
```

⟨Limit meth⟩ マシン単位でのアクセス制限をおこないます. meth には以下のメソッドのいずれかを指定できます.

- **GET** ファイルへのアクセスや CGI プログラムの実行を制限します.

- **POST** POST を用いた CGI プログラムの実行を制限します.

(Limit) 命令で囲まれた間には, 以下の命令が指定できます.

□ **deny**
□ **allow**

アクセスを拒否あるいは許可する WWW クライアントのアドレスを `from` の後に指定します. アドレスは以下のいずれかの形式のものを複数指定できます.

- . から始まるドメイン名 (.osaka-u.ac.jp など)
- マシン名 (es4.es.osaka-u.ac.jp など)
- . で終る IP アドレスの先頭部分 (133.1. など)
- IP アドレス (133.1.136.40 など)
- すべてのマシンを表すキーワード all

□ **order**

deny 命令と allow 命令の評価の順序を指定します.

- **deny,allow** まず deny を評価し, 次に allow を評価します. allow に含まれるマシンからのアクセスを許可することになります.
- **allow,deny** まず allow を評価し, 次に deny を評価します. deny に含まれるマシンからのアクセスを拒否することになります.
- **mutual-failure** allow に含まれるマシンで deny に含まれないマシンからのアクセスを許可します.

□ **require**

利用権を持つユーザ名あるいはユーザグループ名を require entity en1 en2 ... enn の形式で指定します. entity には以下のいずれかを指定することができます.

- **user** en で指定したユーザのみ利用権を得ることができます.
- **group** en で指定したユーザグループに含まれているユーザのみ利用権を得ることができます.
- **valid-user** AuthUserFile で指定したパスワードファイルに定義されているユーザ全員が利用権を得ることができます.

例

```
<Limit GET>
order allow,deny
allow from all
deny from .es.osaka-u.ac.jp
deny from 133.1.136. 133.1.195.
</Limit>
```

この例は, allow, そして deny の順で評価しているので, すべてのマシンからのアクセスを許可しているものの, ドメイン名 .es.osaka-u.ac.jp をもつマシンからのアクセスに対してのみ拒否する, という設定になります. また, IP アドレスからドメイン名を得られないこともありますので, アクセスの拒否に関しては IP アドレスでの指定も合わせて行なう必要があります.

3.5 アクセス制限の方法

実際にどのようにアクセス制限を行なえばよいのかについて, 例を挙げながら説明していくことにします.

3.5.1 限られたマシンからのアクセスのみを許可したい

大阪大学 (ドメイン名 .osaka-u.ac.jp) 以外には公開したくないページがあった場合には, 以下のような設定を行ないます.

例

```
<Directory /var/www/htdocs>
<Limit GET POST>
order deny,allow
deny from all
allow from .osaka-u.ac.jp
</Limit>
</Directory>
```

あるいは

```
<Directory /var/www/htdocs>
<Limit GET POST>
order mutual-failure
allow from .osaka-u.ac.jp
</Limit>
</Directory>
```

しかし, 大阪大学には誰でも利用可能な公開 WWW proxy サーバ (wwwproxy.osaka-u.ac.jp:7080) があり, この proxy サーバを利用してのアクセスはこのマシンの権限でおこなわれます. つまり大阪大学からのアクセスとみなされてしまいます. したがってこのサーバからのアクセスのみを拒否する場合には, 以下のような設定になります.

例

```
<Directory /var/www/htdocs>
<Limit GET POST>
order mutual-failure
allow from .osaka-u.ac.jp
deny from wwwproxy.osaka-u.ac.jp
</Limit>
</Directory>
```

3.5.2 限られたマシンからのアクセスのみを拒否したい

大阪大学に公開したくないページがあった場合には, 以下のような設定を行ないます.

例

```
<Directory /var/www/htdocs>
<Limit GET POST>
order mutual-failure
deny from .osaka-u.ac.jp
allow from all
</Limit>
</Directory>
```

あるいは

```
<Directory /var/www/htdocs>
<Limit GET POST>
order allow,deny
deny from .osaka-u.ac.jp
</Limit>
</Directory>
```

3.5.3 ユーザ名とパスワードでアクセスを制限したい

WWW サーバはクライアントにユーザ名とパスワード入力を要求することによって, 各ディレクトリに対するアクセスを制限することができます.

そのためには, まずパスワードファイルを htpasswd コマンドで作成する必要があります. htpasswd コマンドは, サーバルートの下の support ディレクトリで make htpasswd を実行することによって作成することができます. パスワードファイルは以下のようにして作成します.

```
# touch /var/www/apache_1.1.1/conf/htpasswd
# htpasswd /var/www/apache_1.1.1/conf/htpasswd kawamoto
Adding password for kawamoto
New password: パスワードを入力
Re-type new password: パスワードを入力
```

ここでのユーザ名・パスワードは前述したように, UNIX システムでのアカウントとは無関係です. そのマシンに実在するユーザ名を使用しても構いませんが, UNIX システムのセキュリティ向上のために, 同じパスワードを用いることだけは避けましょう.

また, ユーザグループを定義するグループファイルも必要ならば作成します. グループファイルは以下のようにグループ名とメンバーのユーザ名を並べた行からなるファイルです.

```
groupname: user1 user2 ...
```

以上のようにして, ユーザ名・パスワードファイル・グループを設定し, 次のようにすればアクセス制限を行なうことができます. この例では, Authentication Test という利用権に対して kawamoto というユーザ名を入力しなければならないということと, ユーザ名 kawamoto のパスワード, の 2 点を知っていなければ, このディレクトリに対してアクセスすることはできません.

例

```
<Directory /var/www/htdocs/authtest>
AuthName Authentication Test
AuthType Basic
AuthUserFile /var/www/apache_1.1.1/conf/htpasswd
AuthGroupFile /var/www/apache_1.1.1/conf/htgroup
<Limit GET POST>
require user kawamoto
</Limit>
</Directory>
```

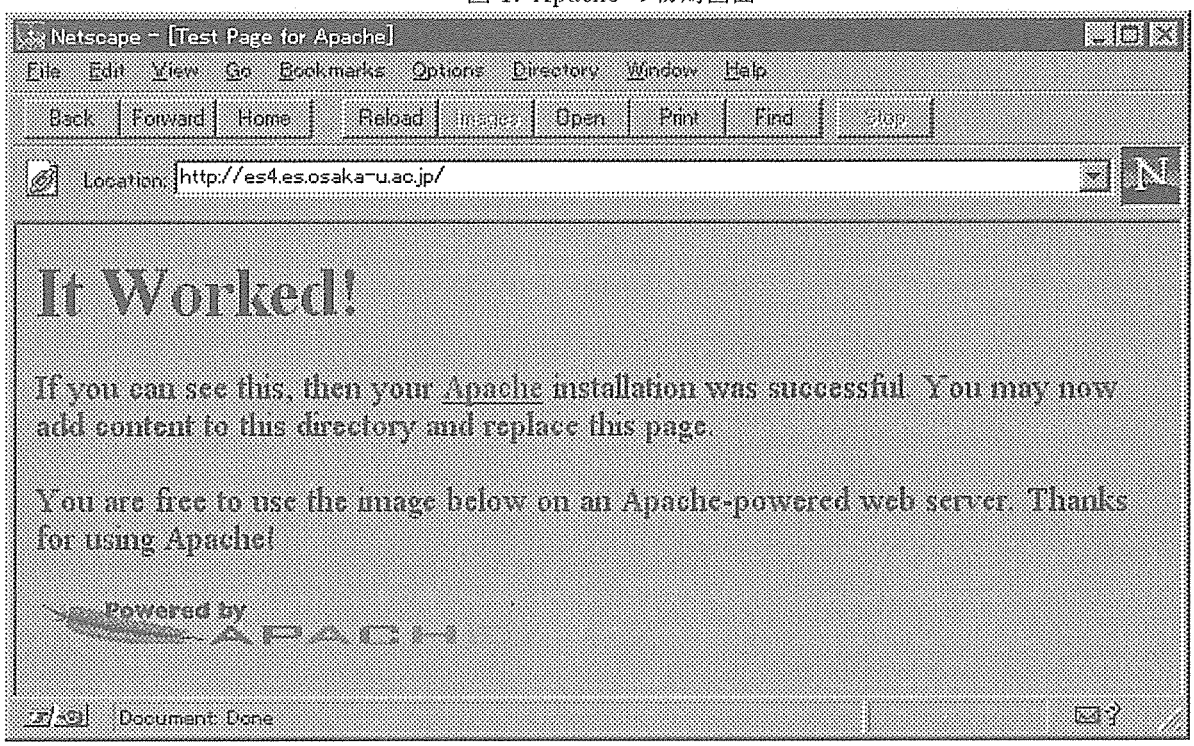
また, require に user ではなく valid-user あるいは group を指定することもできます.valid-user を指定した場合, パスワードファイルに存在するユーザ名なら誰でも利用権を得ることができます. group とグループ名を指定した場合, そのグループのメンバーが利用権を得ることができます.

4 運用

Apache の設定が終ったら, httpd の動作確認をします. httpd のコマンドラインオプションには以下のようなものがあります.

- **-d directory** サーバルートを指定します.

図 1: Apache の初期画面



- **-f file** サーバ設定ファイルがサーバルートの下の `conf/httpd.conf` にない場合に指定します.
- **-v** Apache のバージョンを表示して終了します.

サーバ設定ファイルの ServerType が `standalone` の場合, httpd を以下のようにデーモンとして実行します.

```
$ su
Password:
# /usr/sbin/httpd -d /var/www/apache_1.1.1
```

うまく動作した場合には, WWW クライアントでアクセスすると図 1 のような出力が得られます.

5 Apache についての情報

Apache についての情報は, 以下の Apache についてのページから得ることができます.

⟨URL:http://www.apache.org/⟩

また, 日本では少なくとも以下の ftp サイトから最新版の Apache のソースファイルを手に入れることが可能です.

1. ⟨URL:ftp://ring.aist.go.jp/archives/net/apache/dist/⟩
2. ⟨URL:ftp://ring.asahi-net.or.jp/archives/net/apache/dist/⟩
3. ⟨URL:ftp://ftp.happysize.co.jp/pub/apache/dist/⟩