NAME

mkapachepw − Apache webserver user and group file management utility for Unix-like systems.

SYNOPSIS

mkapachepw.py [-sGUguIicqOohv]

DESCRIPTION

`mkapachepw` is a utility program for creating and maintaining Apache webserver user and group access control files. Its primary value is that it allows large numbers of user and group entries to be managed in separate files. These files are then combined into single, production files for use by the Apache webserver. In this manner, different departments of an organization can independently manage their own user and group information. The webmaster then uses `mkapachepw` to combine them on the production system.

`mkapachepw` has a number of useful features, including:

- Automatically Create Apache Users/Groups From Underlying OS Users/Groups
- Combine User & Group Data From Separately Managed Files Into Single Production Apache Access Control Files
- Specify Which Particular Users/Groups Are To Be Included Or Excluded
- Catch (And Prevent) Redefinition Of User/Groups In Different Files

With `mkapachepw`, you can easily write scripts to automate the creation of custom Apache user and group access control files that contain any/all of the OS users/groups plus any other custom users and groups relevant to your Apache installation. For example, you might have separately managed user/group files for Engineering, Sales, and Marketing. These can then be selectively combined with the users/groups already defined by the underlying operating system.

INSTALLATION

FreeBSD users can simply install the package using the `mkapachepw` port.

All other Unix-like systems require the program to be installed manually:

- Get the distribution tarball from http://www.tundraware.com/Software/mkapachepw
- Copy the ´mkapachepw.py´ file to a directory somewhere in your path.
- Copy the ´mkapachepw.1.gz´ file to the local ´man1´ directory.

OPTIONS

-s suppress inclusion of system user and group information

By default, `mkapachepw` includes the users and groups found in the underlying operating system (starting with the specified starting GID and UID - see the `−g` and `−u` options below). This command line option prevents any system users or groups from being included in the final output files.

-G ´+groupname -groupname +GID -GID ...´
enumerated list of groups to include/exclude

-U ´+username -username +UID -UID ...´
enumerated list of users to include/exclude

By default, `mkapachepw` reads in **all** the users and groups found in the underlying OS databases and any files you've specifically included (see the `-I` and `-i` options below). It then writes these to the output group and user files consistent with the starting GID and UID values specified (see the `-g` and `-u` options below).

You may wish to selectively include or exclude specific groups and/or users in the output files by means of the `-G` and `-U` options respectively. Each of these options takes a quoted list of names and/or IDs as its argument. Each of these groups or users must be prefixed with a `+` symbol to indicate the entry is to be **included** or a − symbol to indicate the entry is to be **excluded.**

For example:

```
mkapachepw.py -I mygroups.inc -i myusers.inc -G "-wheel +admins -45" -U "+bob -fifi"
```

This command reads in all the system groups and users. Then it reads in the group and user entries found in the files, `mygroups.inc` and `myusers.inc`. It then emits Apache group and user files with the group `wheel` and group with a GID of 45 removed, but the group `admins` included. Similarly, user `bob` is included, but user `fifi` is not.

It is probably not immediately obvious why you would ever explicitly include a group or user. After all, the program reads in every single system group/user and all the groups/users found in any included files. Why bother explicitly naming an entry to include? As explained below, it is possible to specify a starting GID or UID when processing the OS group and user databases. Say the starting UID is 100 (the default), but user `bob` has a UID of 99. While `mkapachepw` would initially read this entry in, it would not place it into the output Apache user access control file because this user's UID is below the starting value. By explicitly providing the argument `+bob`, this user will be included in the output despite not satisfying the starting UID test. In other words, the `-G` and `-U` options are the final arbiters of what actually makes it into the output group and user files, regardless of other conditions that may be in effect.

There is another reason why you might explicitly enumerate a user to be included in the output user file. By default, `mkapachepw` does **not** include users in the output that cannot login - i.e. Users whose hashed password is `*`. You can override this by naming such a user explicitly as an argument to the `-U` option.

By combining these options with the starting GID and UID options described below, you can achieve a very fine grain of control of just which groups and users actually end up in a production Apache access control system.

You can repeat these options as many times as you wish on the command line. The argument to each of these options merely needs to be one or more enmeration specifications as described above.

-g number
starting GID to include in group output (default: 100)

-u number

starting UID to include in user output (default: 100)

When group and user information is read from the OS databases, only entries with GID/UID values equal or greater than these starting values are actually included in the output files.

Group and user entries found in any included files (see the `-I` and `-i` options below) are each assigned a "fake" GID/UID of 100000 (this can be changed in the `mkapachepw` program by modifying the `BOGUSID` value). By default, this guarantees that everything in the included files appears in the output. This makes sense, because you probably wouldn't be including the file if you didn't want its content in the output. However, you may wish to selectively include only certain entries in the included files. You can do this by setting the starting GID or UID so high that **nothing** is initially included in the output, and then use enumeration features described above to explicitly name the entries you actually want in the output:

```
mkapachepw.py -I mygroups.inc -g 100001 -G "+admins +webmasters +4433"
```

This would include **all** the system users whose UID was 100 or greater in the user output file (the default action, since no options modifying user processing are specified). It would produce a group output file whose only entries would be the groups `admins`, `webmasters` (regardless whether they were found in the OS group database or in the included file `mygroups.inc`), and the group whose GID was 4433.

-I name of group file to include

-i name of user file to include

These options provide a way to include additional group and user files when producing the final Apache access control files. These files must be in standard Apache format for group and user files respectively. These options can be repeated on the command line to include multiple files:

```
mkapachepw.py -I engineering-groups.inc -I marketing-groups.inc -i engineering-users.inc -i ...
```

`mkapachepw` combines the contents of these files with the content of the OS group and user databases (unless the `-s` option is selected) to create a single, in-memory list of groups and users. The program then "filters" these against the starting GID and UID specification as well as any specific enumerated inclusions or exclusions to produce a final pair of group and user files for use by Apache.

Any files you request to be included will be processed **after** `mkapachepw` reads the OS group and user databases. This means that, if one of your included files has a group/user name that is the same as one of the system groups/user, you will get a "collision". Unless you prohibit collisions (see the `-c` option below), this means the entry from the included file will supercede the OS entry (though you will be warned about the collision). Say, for example, you have user named `mary` both in your OS and in one of your included files. Then, the password specified for `mary` in the **included file** will end up in the final Apache user access control file, not the password found in the OS user database.

Comments and blank lines found in any included file are removed. After `mkapachepw` has combined all the entries found in the OS databases and all the included files, and decided which ones to actually emit into the output files, it sorts them into alphabetic order.

It's important to realize that `mkapachepw` does no "sanity" checking on included files. It presumes them to be in correct Apache format for group and user access control files respectively. If you include a file that is not in this format, the program will probably blow up spectacularly, and write garbage output files, or nothing at all.

-c prohibit collisions

Ordinarily, if an included file has the same group or user name as one found in the underlying OS databases, `mkapachepw` merely warns you of this "collision", accepts the values found in the included file in place of the system values, and continues. If you select this option, the program will strictly prohibit collisions and exit immediately when it detects one.

-q quiet mode

This inhibits the display of collision warnings.

-O group filename

Names the file to which group information will be written. (default: `./.htgroups`). The file is produced with read permission for the owner only. Before installing the file, be sure to change the owner/group for this file as appropriate for your Apache installation.

If you specify − here, group output is written to stdout.

-o user filename

Names the file to which user information will be written. (default: `./.htusers`). The file is produced with read permission for the owner only. Before installing the file, be sure to change the owner/group for this file as appropriate for your Apache installation.

If you specify − here, group output is written to stdout.

-h print help information

-v print detailed program version information

EXAMPLES

```
mkapachepw.py
```

Output files will contain only system groups and users whose GIDs/UIDs are 100 or higher.

```
mkapachepw.py -c -I groups.inc
```

Output files will contain system groups and user whose GIDs/UIDs are 100 or higher, and all the groups found in `groups.inc`. If any of the groups in this included file has the same name as a system group, the program will emit an error and halt, writing no output.

```
mkapachepw.py -G "-wheel -bin" -U "-root -bin"
```

The output files will contain all system groups and users whose GIDs/UIDs are 100 or higher **except** the groups `wheel` and `bin` and the users `root` and `bin`, which will not appear in the final group and user files respectively.

```
mkapachepw.py -s -I mygroups.inc -i myusers.inc
```

Only the entries in the included files will be processed. No system groups or users will be present in the final output.

```
mkapachepw.py -g 100001 -u 100001 -I ... -i ... -G ... -U ...
```

Nothing is included by default. Only the entries explicitly enumerated by the `-G` and `-U` options will be included from both the system and included file entries.

EXIT CODES

```
0       No error, normal completion

1       Invalid command line:
          - Cannot open include file
          - Invalid command line option
          - Extraneous argument(s) after command line options
          - Invalid argument for starting GID or UID

2       Invalid enumeration argument:
          - Enumerated entry missing an inclusion/exclusion prefix (+/-)
          - Attempt to include/exclude a non-existent GID, UID, or name

3       Cannot open output file

4       Included file attempted to redefine existing user or group,
        and collisions are currently forbidden via the -c option
```

OTHER NOTES

`mkapachepw` is a pure-Python module and should run anywhere a recent Python implementation is found. However, it is Unix-specific in that it requires the ´pwd´ and ´grp´ modules for accessing the underlying OS user and group databases. Note that access to these system databases requires the program be run by the root user.

In order for this program to produce correct access control files, Apache and the underlying OS must agree on the hash algorithm used to encrypt passwords. This should be the default case in most instances.

This program has only been tested on FreeBSD, though it should work on other Unix-like variants.

The program will not run on Win32 systems.

Output files are "stamped" with comments containing the program version, the time, the date, and the command line that produced that file.

BUGS AND MISFEATURES

None known as of this release.

COPYRIGHT AND LICENSING

AUTHOR

```
Tim Daneliuk
mkapachepw@tundraware.com
```

DOCUMENT REVISION INFORMATION

$Id: mkapachepw.1,v 1.107 2005/04/12 18:49:36 toor Exp $