



Linux Ubuntu 26.04 Manual Pages on command 'cvtsudoers.ws.1'

\$ man cvtsudoers.ws.1

CVTSUDOERS(1) General Commands Manual CVTSUDOERS(1)

NAME

cvtsudoers ? convert between sudoers file formats

SYNOPSIS

```
cvtsudoers [-ehMpV] [-b dn] [-c conf_file] [-d deftypes] [-f output_format]
           [-i input_format] [-l increment] [-l log_file] [-m filter] [-o output_file]
           [-O start_point] [-P padding] [-s sections] [input_file ...]
```

DESCRIPTION

The `cvtsudoers` utility accepts one or more security policies in either sudoers or LDIF format as input, and generates a single policy of the specified format as output. The default input format is sudoers. The default output format is LDIF. It is only possible to convert a policy file that is syntactically correct.

If no `input_file` is specified, or if it is `?-?`, the policy is read from the standard input.

Input files may be optionally prefixed with a host name followed by a colon (`?:?`) to make the policy rules specific to a host when merging multiple files. By default, the result is written to the standard output.

The options are as follows:

`-b dn, --base=dn`

The base DN (distinguished name) that will be used when performing LDAP queries.

Typically this is of the form `?ou=SUDOers,dc=my-domain,dc=com?` for the domain `my-do?`

`main.com`. If this option is not specified, the value of the `SUDOERS_BASE` environment

variable will be used instead. Only necessary when converting to LDIF format.

`-c conf_file, --config=conf_file`

Specify the path to the configuration file. Defaults to /etc/cvtsudoers.conf.

-d deftypes, --defaults=deftypes

Only convert Defaults entries of the specified types. One or more Defaults types may be specified, separated by a comma (`,`). The supported types are:

all All Defaults entries.

global Global Defaults entries that are applied regardless of user, runas, host, or command.

user Per-user Defaults entries.

runas Per-runas user Defaults entries.

host Per-host Defaults entries.

command Per-command Defaults entries.

See the Defaults section in sudoers(5) for more information.

If the -d option is not specified, all Defaults entries will be converted.

-e, --expand-aliases

Expand aliases in input_file. Aliases are preserved by default when the output format is JSON or sudoers.

-f output_format, --output-format=output_format

Specify the output format (case-insensitive). The following formats are supported:

CSV CSV (comma-separated value) files are often used by spreadsheets and report generators. See ?CSV output format? for more details.

JSON JSON (JavaScript Object Notation) files are usually easier for third-party applications to consume than the traditional sudoers format. The various values have explicit types which removes much of the ambiguity of the sudoers format. See ?JSON output format? for more details.

LDIF LDIF (LDAP Data Interchange Format) files can be imported into an LDAP server for use with sudoers.Ldap(5).

Conversion to LDIF has the following limitations:

? Command, host, runas, and user-specific Defaults lines cannot be translated as they don't have an equivalent in the sudoers LDAP schema.

? Command, host, runas, and user aliases are not supported by the sudoers LDAP schema so they are expanded during the conversion.

sudoers Traditional sudoers format. A new sudoers file will be reconstructed from the parsed input file. Comments are not preserved and data from any in?

clude files will be output inline.

`--group-file=file`

When the `-M` option is also specified, perform group queries using `file` instead of the system group database.

`-h, --help`

Display a short help message to the standard output and exit.

`-i input_format, --input-format=input_format`

Specify the input format. The following formats are supported:

LDIF LDIF (LDAP Data Interchange Format) files can be exported from an LDAP server to convert security policies used by `sudoers.ldap(5)`. If a base DN (distinguished name) is specified, only `sudoRole` objects that match the base DN will be processed. Not all `sudoOptions` specified in a `sudoRole` can be translated from LDIF to `sudoers` format.

sudoers Traditional `sudoers` format. This is the default input format.

`-I increment, --increment=increment`

When generating LDIF output, increment each `sudoOrder` attribute by the specified number. Defaults to an increment of 1.

`-l log_file, --logfile=log_file`

Log conversion warnings to `log_file` instead of to the standard error. This is particularly useful when merging multiple `sudoers` files, which can generate a large number of warnings.

`-m filter, --match=filter`

Only output rules that match the specified filter. A filter expression is made up of one or more `key = value` pairs, separated by a comma (`?,?`). The key may be `?cmdnd?` (or `?cmd?`), `?host?`, `?group?`, or `?user?`. For example, `user = operator` or `host = www`. An upper-case `Cmnd_Alias`, `Host_alias`, or `User_Alias` may be specified as the `?cmdnd?`, `?host?`, or `?user?`.

A matching `sudoers` rule may also include users, groups, and hosts that are not part of the filter. This can happen when a rule includes multiple users, groups, or hosts. To prune out any non-matching user, group, or host from the rules, the `-p` option may be used.

By default, the password and group databases are not consulted when matching against the filter so the users and groups do not need to be present on the local system

(see the -M option). Only aliases that are referenced by the filtered policy rules will be displayed.

-M, --match-local

When the -m option is also specified, use password and group database information when matching users and groups in the filter. Only users and groups in the filter that exist on the local system will match, and a user's groups will automatically be added to the filter. If the -M is not specified, users and groups in the filter do not need to exist on the local system, but all groups used for matching must be explicitly listed in the filter.

-o output_file, --output=output_file

Write the converted output to output_file. If no output_file is specified, or if it is ?-?, the converted sudoers policy will be written to the standard output.

-O start_point, --order-start=start_point

When generating LDIF output, use the number specified by start_point in the sudoOrder attribute of the first sudoRole object. Subsequent sudoRole object use a sudoOrder value generated by adding an increment, see the -I option for details. Defaults to a starting point of 1. A starting point of 0 will disable the generation of sudoOrder attributes in the resulting LDIF file.

--passwd-file=file

When the -M option is also specified, perform passwd queries using file instead of the system passwd database.

-p, --prune-matches

When the -m option is also specified, cvtsudoers will prune out non-matching users, groups, and hosts from matching entries.

-P padding, --padding=padding

When generating LDIF output, construct the initial sudoOrder value by concatenating order_start and increment, padding the increment with zeros until it consists of padding digits. For example, if order_start is 1027, padding is 3, and increment is 1, the value of sudoOrder for the first entry will be 1027000, followed by 1027001, 1027002, etc. If the number of sudoRole entries is larger than the padding would allow, cvtsudoers will exit with an error. By default, no padding is performed.

-s sections, --suppress=sections

Suppress the output of specific sections of the security policy. One or more sec?

tion names may be specified, separated by a comma (`,`). The supported section name are: defaults, aliases and privileges (which may be shortened to privs).

`-V, --version`

Print the cvtsudoers and sudoers grammar versions and exit.

Merging multiple files

When multiple input files are specified, cvtsudoers will attempt to merge them into a single policy file. It is assumed that user and group names are consistent among the policy files to be merged. For example, user `?bob?` on one host is the same as user `?bob?` on another host.

When merging policy files, it is possible to prefix the input file name with a host name, separated by a colon (`?:?`). When the files are merged, the host name will be used to restrict the policy rules to that specific host where possible.

The merging process is performed as follows:

- ? Each input file is parsed into internal sudoers data structures.
- ? Aliases are merged and renamed as necessary to avoid conflicts. In the event of a conflict, the first alias found is left as-is and subsequent aliases of the same name are renamed with a numeric suffix separated with an underscore (`?_?`). For example, if there are two different aliases named `SERVERS`, the first will be left as-is and the second will be renamed `SERVERS_1`. References to the renamed alias are also updated in the policy file. Duplicate aliases (those with identical contents) are pruned.
- ? Defaults settings are merged and duplicates are removed. If there are conflicts in the Defaults settings, a warning is emitted for each conflict. If a host name is specified with the input file, cvtsudoers will change the global Defaults settings in that file to be host-specific. A warning is emitted for command, user, or runas-specific Defaults settings which cannot be made host-specific.
- ? Per-user rules are merged and duplicates are removed. If a host name is specified with the input file, cvtsudoers will change rules that specify a host name of `ALL` to the host name associated with the policy file being merged. The merging of rules is currently fairly simplistic but will be improved in a later release.

It is possible to merge policy files with differing formats.

The cvtsudoers.conf file

Options in the form `?keyword = value?` may also be specified in a configuration file, `/etc/cvtsudoers.conf` by default. The following keywords are recognized:

defaults = deftypes

See the description of the -d command line option.

expand_aliases = yes | no

See the description of the -e command line option.

group_file = file

See the description of the --group-file command line option.

input_format = ldif | sudoers

See the description of the -i command line option.

match = filter

See the description of the -m command line option.

match_local = yes | no

See the description of the -M command line option.

order_increment = increment

See the description of the -I command line option.

order_start = start_point

See the description of the -O command line option.

output_format = csv | json | ldif | sudoers

See the description of the -f command line option.

padding = padding

See the description of the -P command line option.

passwd_file = file

See the description of the --passwd-file command line option.

prune_matches = yes | no

See the description of the -p command line option.

sudoers_base = dn

See the description of the -b command line option.

suppress = sections

See the description of the -s command line option.

Options on the command line will override values from the configuration file.

JSON output format

The sudoers JSON format may contain any of the following top-level objects:

Defaults

An array of objects, each containing an Options array and an optional Binding array.

The Options array consists of one or more objects, each containing a ?name:value? pair that corresponds to a sudoers Defaults setting. Options that operate on a list will also include an operation entry in the object, with a value of ?list_assign? for ?=?, ?list_add? for ?+=?, or ?list_remove? for ?-=?.

The optional Binding array consists of one or more objects, each containing a ?name:value? pair and an optional negated entry, which will negate any comparison performed with the object. If a Binding is present, the setting will only take effect if one of the specified command, hostname, netgroup, networkaddr, nonunixgid, nonunixgroup, usergid, usergroup, userid, username, or alias entries match.

For example, the following sudoers entry:

```
Defaults@somehost set_home, env_keep += DISPLAY
```

converts to:

```
"Defaults": [
  {
    "Binding": [
      { "hostname": "somehost" }
    ],
    "Options": [
      { "set_home": true },
      {
        "operation": "list_add",
        "env_keep": [
          "DISPLAY"
        ]
      }
    ]
  }
]
```

User_Aliases

A JSON object containing one or more sudoers User_Alias entries where each named alias has as its value an array containing one or more objects. Each object contains a ?name:value? pair and an optional negated entry, which will negate any comparison performed with the object. The name may be one of netgroup, nonunixgid, nonunixgroup,

useralias, usergid, usergroup, userid, or username.

For example, the following sudoers entry:

User_Alias SYSADMIN = will, %wheel, +admin

converts to:

```
"User_Aliases": {  
  "SYSADMIN": [  
    { "username": "will" },  
    { "usergroup": "wheel" },  
    { "netgroup": "admin" }  
  ]  
}
```

Runas_Aliases

A JSON object containing one or more sudoers Runas_Alias entries, where each named alias has as its value an array containing one or more objects. Each object contains a ?name:value? pair and an optional negated entry, which will negate any comparison performed with the object. The name may be one of netgroup, nonunixgid, nonunixgroup, runasalias, usergid, usergroup, userid, or username.

For example, the following sudoers entry:

Runas_Alias DB = oracle, sybase : OP = root, operator

converts to:

```
"Runas_Aliases": {  
  "DB": [  
    { "username": "oracle" },  
    { "username": "sybase" }  
  ],  
  "OP": [  
    { "username": "root" },  
    { "username": "operator" }  
  ]  
}
```

Host_Aliases

A JSON object containing one or more sudoers Host_Alias entries where each named alias has as its value an array containing one or more objects. Each object contains a

?name:value? pair and an optional negated entry, which will negate any comparison per?
formed with the object. The name may be one of hostalias, hostname, netgroup, or
networkaddr.

For example, the following sudoers entries:

Host_Alias DORMNET = 128.138.243.0, 128.138.204.0/24

Host_Alias SERVERS = boulder, refuge

convert to:

```
"Host_Aliases": {  
  "DORMNET": [  
    { "networkaddr": "128.138.243.0" },  
    { "networkaddr": "128.138.204.0/24" }  
  ],  
  "SERVERS": [  
    { "hostname": "boulder" },  
    { "hostname": "refuge" }  
  ]  
}
```

Cmnd_Aliases

A JSON object containing one or more sudoers Cmnd_Alias entries where each named alias
has as its value an array containing one or more objects. Each object contains a
?name:value? pair and an optional negated entry, which will negate any comparison per?
formed with the object. The name may be either another cmndalias or a command. For
example, the following sudoers entries:

Cmnd_Alias SHELLS = /bin/bash, /bin/csh, /bin/sh, /bin/zsh

Cmnd_Alias VIPW = /usr/bin/chpass, /usr/bin/chfn, /usr/bin/chsh, \
/usr/bin/passwd, /usr/sbin/vigr, /usr/sbin/vipw

convert to:

```
"Cmnd_Aliases": {  
  "SHELLS": [  
    { "command": "/bin/bash" },  
    { "command": "/bin/csh" },  
    { "command": "/bin/sh" },  
    { "command": "/bin/zsh" }  
  ]  
}
```

```

],
"VIPW": [
  { "command": "/usr/bin/chpass" },
  { "command": "/usr/bin/chfn" },
  { "command": "/usr/bin/chsh" },
  { "command": "/usr/bin/passwd" },
  { "command": "/usr/sbin/vigr" },
  { "command": "/usr/sbin/vipw" }
]
}

```

User_Specs

A JSON array containing one or more objects, each representing a sudoers User_Spec. Each object in the User_Specs array should contain a User_List array, a Host_List array and a Cmnd_Specs array.

A User_List consists of one or more objects. Each object contains a ?name:value? pair and an optional negated entry, which will negate any comparison performed with the object. The name may be one of netgroup, nonunixgid, nonunixgroup, useralias, usergid, usergroup, userid, or username. If username is set to the special value ALL, it will match any user.

A Host_List consists of one or more objects. Each object contains a ?name:value? pair and an optional negated entry, which will negate any comparison performed with the object. The name may be one of hostalias, hostname, netgroup, or networkaddr. If hostname is set to the special value ALL, it will match any host.

The Cmnd_Specs array consists of one or more JSON objects describing a command that may be run. Each Cmnd_Specs is made up of a Commands array, an optional runasusers array, an optional runasgroups array, and an optional Options array.

The Commands array consists of one or more objects containing ?name:value? pair elements. The following names and values are supported:

command A string containing the command to run. The special value ALL it will match any command.

negated A boolean value that, if true, will negate any comparison performed with the object.

sha224 One or more SHA224 digests for the command in string form. Multiple digests

of the same type are stored as an array.

sha256 One or more SHA256 digests for the command in string form. Multiple digests of the same type are stored as an array.

sha384 One or more SHA384 digests for the command in string form. Multiple digests of the same type are stored as an array.

sha512 One or more SHA512 digests for the command in string form. Multiple digests of the same type are stored as an array.

The `runasusers` array consists of objects describing users the command may be run as. Each object contains a `?name:value?` pair and an optional `negated` entry, which will negate any comparison performed with the object. The name may be one of `netgroup`, `nonunixgid`, `nonunixgroup`, `runasalias`, `usergid`, `usergroup`, `userid`, or `username`. If `username` is set to the special value `ALL`, it will match any user. If `username` is set to the empty string `??`, it will match the invoking user.

The `runasgroups` array consists of objects describing groups the command may be run as. Each object contains a `?name:value?` pair and an optional `negated` entry, which will negate any comparison performed with the object. The name may be one of `runasalias`, `usergid`, or `usergroup`. If `usergroup` is set to the special value `ALL`, it will match any group.

The `Options` array is of the same format as the one in the `Defaults` object. Any `Tag_Spec` entries in `sudoers` are converted to `Options`. A user with `?sudo ALL? privi?leges` will automatically have the `setenv` option enabled to match the implicit behavior provided by `sudoers`.

For example, the following `sudoers` entry:

```
millert ALL = (ALL : ALL) NOPASSWD: ALL, !/usr/bin/id
```

converts to:

```
"User_Specs": [
  {
    "User_List": [
      { "username": "millert" }
    ],
    "Host_List": [
      { "hostname": "ALL" }
    ],
  },
]
```

```

"Cmnd_Specs": [
  {
    "runasusers": [
      { "username": "ALL" }
    ],
    "runasgroups": [
      { "usergroup": "ALL" }
    ],
    "Options": [
      { "authenticate": false },
      { "setenv": true }
    ],
    "Commands": [
      { "command": "ALL" },
      {
        "command": "/usr/bin/id",
        "negated": true
      }
    ]
  }
]

```

CSV output format

CSV (comma-separated value) files are often used by spreadsheets and report generators. For CSV output, `cvtsudoers` double quotes strings that contain commas. For each literal double quote character present inside the string, two double quotes are output. This method of quoting commas is compatible with most spreadsheet programs.

There are three possible sections in `cvtsudoers`'s CSV output, each separated by a blank line:

defaults

This section includes any Defaults settings in `sudoers`. The defaults section begins with the following heading:

defaults_type, binding, name, operator, value

The fields are as follows:

defaults_type

The type of Defaults setting; one of defaults, defaults_command, defaults_host, defaults_runas, or defaults_user.

binding

For defaults_command, defaults_host, defaults_runas, and defaults_user this is the value that must match for the setting to be applied.

name The name of the Defaults setting.

operator

The operator determines how the value is applied to the setting. It may be either == (assignment), += (append), or -= (remove).

value

The setting's value, usually a string or, for settings used in a boolean context, true or false.

aliases

This section includes any Cmnd_Alias, Host_Alias, Runas_Alias, or User_Alias, entries from sudoers. The aliases section begins with the following heading:

alias_type, alias_name, members

The fields are as follows:

alias_type

The type of alias; one of Cmnd_Alias, Host_Alias, Runas_Alias, or User_Alias.

alias_name

The name of the alias; a string starting with an upper-case letter that consists of upper-case letters, digits, or underscores.

members

A comma-separated list of members belonging to the alias. Due to the use of commas, members is surrounded by double quotes if it contains more than one member.

rules

This section includes the sudoers rules that grant privileges. The rules section begins with the following heading:

rule, user, host, runusers, rungroups, options, command

The fields are as follows:

rule This field indicates a sudoers rule entry.

user The user the rule applies to. This may also be a Unix group (preceded by a `??` character), a non-Unix group (preceded by `?:?`) or a netgroup (preceded by a `?+?` character) or a `User_Alias`. If set to the special value `ALL`, it will match any user.

host The host the rule applies to. This may also be a netgroup (preceded by a `?+?` character) or a `Host_Alias`. If set to the special value `ALL`, it will match any host.

runusers

An optional comma-separated list of users (or `Runas_Aliases`) the command may be run as. If it contains more than one member, the value is surrounded by double quotes. If set to the special value `ALL`, it will match any user. If empty, the root user is assumed.

rungroups

An optional comma-separated list of groups (or `Runas_Aliases`) the command may be run as. If it contains more than one member, the value is surrounded by double quotes. If set to the special value `ALL`, it will match any group. If empty, the runuser's group is used.

options

An optional list of Defaults settings to apply to the command. Any `Tag_Spec` entries in sudoers are converted to options.

commands

A list of commands, with optional arguments, that the user is allowed to run. If set to the special value `ALL`, it will match any command.

For example, the following sudoers entry:

```
millert ALL = (ALL : ALL) NOPASSWD: ALL, !/usr/bin/id
```

converts to:

```
rule,millert,ALL,ALL,ALL,"!authenticate","ALL,!/usr/bin/id"
```

FILES

`/etc/cvtsudoers.conf` default configuration for cvtsudoers

EXAMPLES

Convert `/etc/sudoers` to LDIF (LDAP Data Interchange Format) where the `ldap.conf` file uses a

sudoers_base of my-domain,dc=com, storing the result in sudoers.ldif:

```
$ cvtsudoers -b ou=SUDOers,dc=my-domain,dc=com -o sudoers.ldif \
/etc/sudoers
```

Convert /etc/sudoers to JSON format, storing the result in sudoers.json:

```
$ cvtsudoers -f json -o sudoers.json /etc/sudoers
```

Parse /etc/sudoers and display only rules that match user ambrose on host hastur:

```
$ cvtsudoers -f sudoers -m user=ambrose,host=hastur /etc/sudoers
```

Same as above, but expand aliases and prune out any non-matching users and hosts from the expanded entries.

```
$ cvtsudoers -ep -f sudoers -m user=ambrose,host=hastur /etc/sudoers
```

Convert sudoers.ldif from LDIF to traditional sudoers format:

```
$ cvtsudoers -i ldif -f sudoers -o sudoers.new sudoers.ldif
```

Merge a global sudoers file with two host-specific policy files from the hosts ?xyzz? and ?plugh?:

```
$ cvtsudoers -f sudoers -o sudoers.merged sudoers \
xyzz:sudoers.xyzz plugh:sudoers.plugh
```

SEE ALSO

sudoers(5), sudoers.ldap(5), sudo(8)

AUTHORS

Many people have worked on sudo over the years; this version consists of code written primarily by:

Todd C. Miller

See the CONTRIBUTORS.md file in the sudo distribution (<https://www.sudo.ws/about/contributors/>) for an exhaustive list of people who have contributed to sudo.

BUGS

If you believe you have found a bug in cvtsudoers, you can either file a bug report in the sudo bug database, <https://bugzilla.sudo.ws/>, or open an issue at <https://github.com/sudo-project/sudo/issues>. If you would prefer to use email, messages may be sent to the sudo-workers mailing list, <https://www.sudo.ws/mailman/listinfo/sudo-workers> (public) or [<sudo@sudo.ws>](mailto:sudo@sudo.ws) (private).

Please do not report security vulnerabilities through public GitHub issues, Bugzilla or mailing lists. Instead, report them via email to [<Todd.Miller@sudo.ws>](mailto:Todd.Miller@sudo.ws). You may encrypt your message with PGP if you would like, using the key found at <https://www.sudo.ws/dist/PGP?>

PKEYS.

SUPPORT

Limited free support is available via the sudo-users mailing list, see <https://www.sudo.ws/mailman/listinfo/sudo-users> to subscribe or search the archives.

DISCLAIMER

cvtsudoers is provided "AS IS" and any express or implied warranties, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose are disclaimed. See the LICENSE.md file distributed with sudo or <https://www.sudo.ws/about/license/> for complete details.

Sudo 1.9.17p2

February 22, 2025

CVTSUDOERS(1)