NAME

id — get credentials

SYNOPSIS

id [*user*]...
id **-u**|**-g**|**-G** [**-nrz**] [*user*]...
id **-Z** [**-z**]
groups [*user*]...
whoami

DESCRIPTION

The flag-free form writes *user*s' (each of which can either be a name or a numeric ID; if none specified — the process') credentials to the standard output stream:

uid=0(root) **gid=**0(root) **groups=**0(root)**,10**(wheel)**,**22
context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
uid=1000(cicada) **gid=**100(users) **euid=**0(root) **egid=**39(irc) **groups=**100(users)**,**4(adm)**,**27(sudo)

When an ID is missing from `passwd`(5) or `group`(5) its name and parentheses are omitted.

The IDs listed are real; **euid=** and **egid=**, containing the effective IDs, are only written if they differ.

If one of **-ugGZ** is specified, the IDs are effective by default, and only the specified field is output, numerically. For example, those very same invocations, but with **-G** and **-u** respectively:

```
0 10 22
0
```

And with **-rn**:

```
root wheel 22
cicada
```

In this mode, unknown groups are written as numbers and a diagnostic is issued.

The set of supplementary groups is sorted, but always starts with the primary group.

An unknown *user* is not a fatal error — a diagnostic is issued and processing continues.

The mandatory access control context is omitted from the default format if unavailable or unset, and **-Z** is forbidden if it's unavailable; additionally, as it's a property of the process, not user, it's only available if no *user* was specified.

groups is equivalent to **id -Gn**.
whoami is equivalent to **id -un**.

OPTIONS

-u, **--user**	Write just the user ID.
-g, **--group**	Write just the primary group ID.
-G, **--groups**	Write just the secondary group IDs.
-Z, **--context**	Write just the security context.
-n, **--name**	Write the specified field as a human-readable name, rather than numerical ID.
-r, **--real**	Use the real user or primary group ID of the process, rather than the effective one. No effect with *user*s.
-z, **--zero**	Terminate each entry with a NUL character, rather than a single space or newline.
-a	Ignored for compatibility with the illumos gate, whose default **id** requires it to write the supplementary groups. (But the version from AT&T research dated 2004 also just ignores it.)

EXIT STATUS

1 if at least one *user* or group wasn't on file.

SEE ALSO

who(1), getegid(2), getgroups(2), getuid(2), getgrouplist(3), getgwgid(3), getpwuid(3), selinux(8)

STANDARDS

id conforms to IEEE Std 1003.1-2024 (“POSIX.1”).

Mandatory access control behaviour (**-Z**, **context=**) is compatible with the GNU system, though the latter deviates from the standard if a context is present; this is not considered a problem, as **-ugG** is recommended by the standard for noninteractive use, and scripts relying on the default CB-UNIX-style output already filter it.

-za and multiple *user*s are extensions, also present on the GNU system.

The ordering of supplementary groups is not guaranteed.

HISTORY

whoami first appeared in 2BSD as whoami(UCB), noting:

> It works even if you are su’d, while ’who am i’ does not since it uses /etc/utmp.

groups appeared in 4.1cBSD as groups(1):

> groups − show group memberships

with no arguments and present-day output, which also described the system’s general group handling.

4.2BSD allows one *user*, as present-day.

id first appeared in CB-UNIX at or before version 2.1, using the familiar

> **uid=**1000(user) **gid=**100(users) **euid=**0(root) **egid=**0(wheel)

format, with the notable exception of auxilliary groups, which originate from the BSD. CB-UNIX was, among others, the basis for AT&T System III UNIX, where it first saw light outside AT&T.

4.4BSD was the first to include **id**, quoting conformance with IEEE Std 1003.2 (“POSIX.2”), including the

> **groups=**100(users)**,**0(wheel)

entry and **-p** extension, deprecating **groups** and **whoami**, replacing them with shell wrappers.

Since coreutils 8.31 (2019-03-10), the GNU system allows multiple *user*s for **id**, too.