ONE IDENTITY

by Quest®

What's new in sudo 1.9

Peter Czanik
Open Source Evangelist
One Identity
@PCzanik

Overview

- What is sudo
- From aliases to plugins
- Alerting with syslog-ng
- What is new in 1.9?

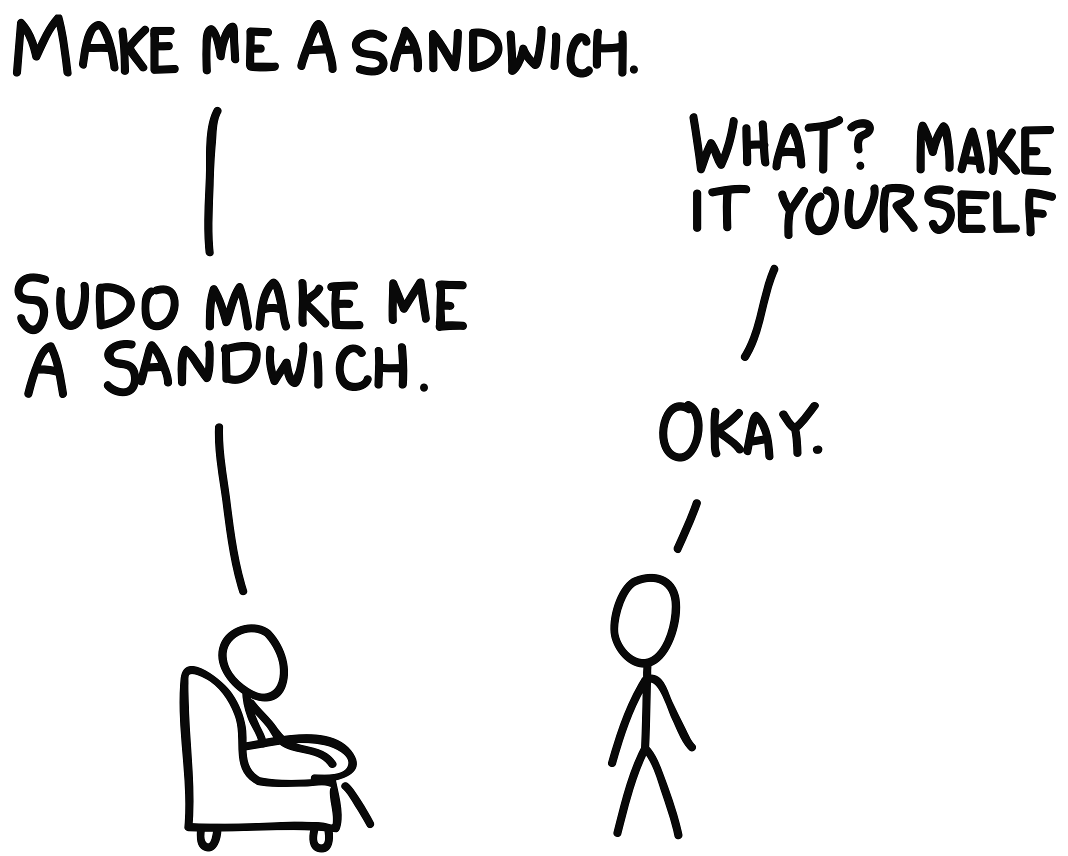
What is sudo?

- Answers, depending on experience and size of environment:
  - A tool to complicate life
  - A prefix for administrative commands
  - A way to see who did what

What is sudo?

- Sudo allows a system administrator to delegate authority by giving certain users the ability to run some commands as root or another user while providing an audit trail of the commands and their arguments. (https://www.sudo.ws/)
- A lot more, than just a prefix

It can make you a sandwich… (by XKCD)



ONE IDENTITY by Quest

Basic /etc/sudoers

```
%wheel        ALL=(ALL)        ALL
```

- Who
- Where
- As which user
- Which command

Aliases

- Simplify configuration
- Less error-prone

```
Host_Alias    WEBSERVERS = www1, www2, www3
User_Alias    ADMINS = smith, johnson, williams
Cmnd_Alias    REBOOT = /sbin/halt, /sbin/reboot, /sbin/poweroff

ADMINS   WEBSERVERS = REBOOT
```

Defaults

- Changes the default behavior:

```
Defaults secure_path="/usr/sbin:/usr/bin:/sbin:/bin"
Defaults env_keep = "LANG LC_ADDRESS LC_CTYPE"
Defaults !insults
```

- Can be user/host/etc specific

```
Defaults:%wheel insults
```

ONE IDENTITY by Quest

Insults

- Fun, but not always PC :)

```
czanik@linux-mewy:~> sudo ls
[sudo] password for root:
Hold it up to the light --- not a brain in sight!
[sudo] password for root:
My pet ferret can type better than you!
[sudo] password for root:
sudo: 3 incorrect password attempts
czanik@linux-mewy:~>
```

Digest verification

```
peter ALL =
sha244:11925141bb22866afdf257ce7790bd6275feda80b
3b241c108b79c88 /usr/bin/passwd
```

- Modified binaries do not run
- Difficult to maintain
- Additional layer of protection

Session recording

- Recording the terminal
- Playback
- Difficult to modify (not cleartext)
- Easy to delete (saved locally) with unlimited access
    - Stay tuned :)

Plugin-based architecture

- Starting with version 1.8
- Replace or extend functionality
- Both open source and commercial

Configuration hints

- Use visudo for syntax check
- Use EDITOR to use another text editor :)
- A syntactically correct config still does not mean that you can execute anything :)
- root password (even for Ubuntu!)

Configuration

- Read from top to bottom
- Start with generic
- Add exceptions at the end

Sample Configuration

```
Defaults   !visiblepw
Defaults    always_set_home
Defaults    match_group_by_gid
Defaults    always_query_group_plugin
Defaults    env_reset
Defaults    env_keep =  "COLORS DISPLAY HOSTNAME HISTSIZE KDEDIR LS_COLORS"
Defaults    env_keep += "MAIL PS1 PS2 QTDIR USERNAME LANG LC_ADDRESS LC_CTYPE"
Defaults    secure_path = /sbin:/bin:/usr/sbin:/usr/bin
root   ALL=(ALL)           ALL
%wheel          ALL=(ALL)           ALL
Defaults:%wheel insults
Defaults !insults
Defaults log_output
```

Where is the problem?

(There was a common mistake)

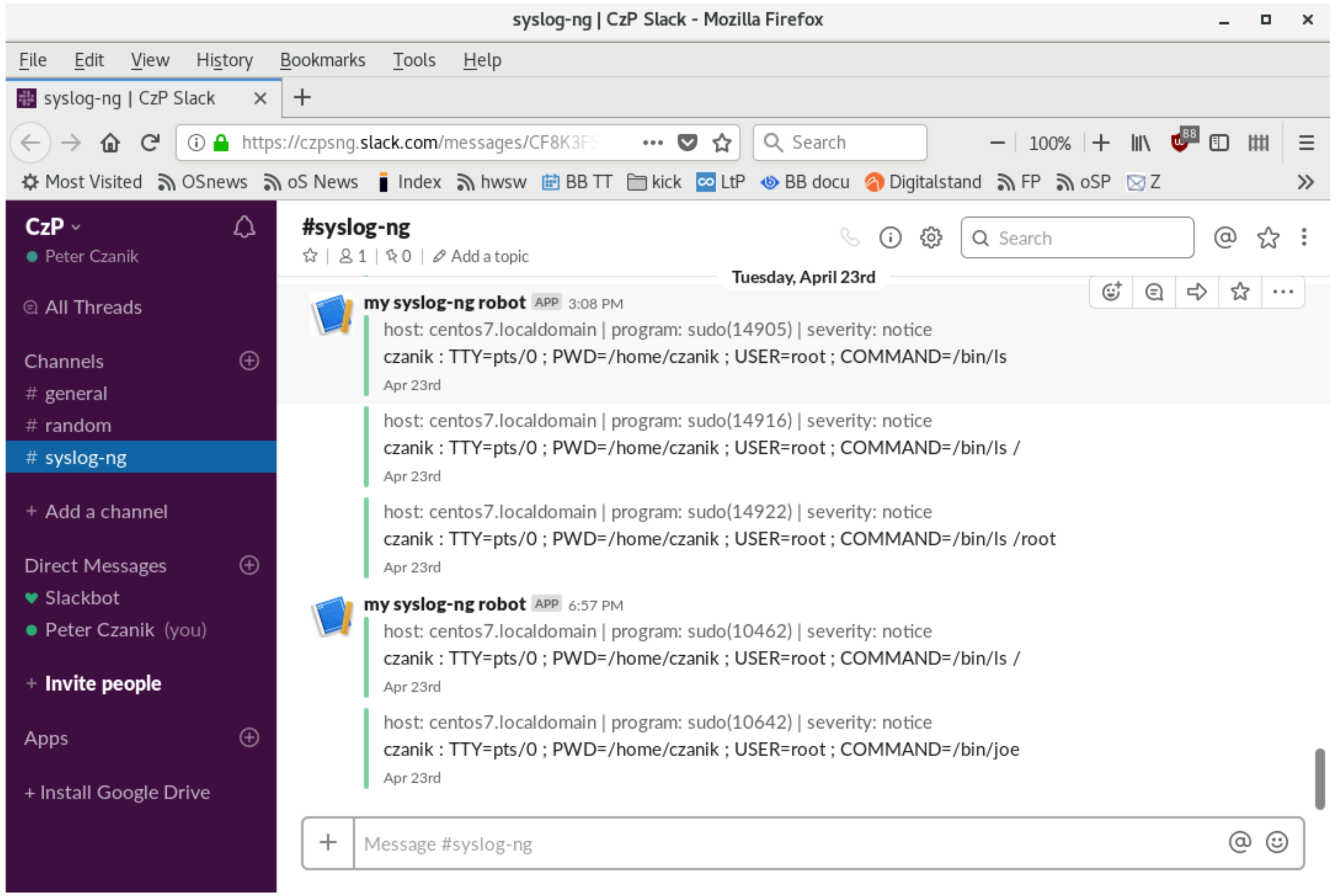
Central management

- Puppet, Ansible, etc.
  - Not real-time
  - Users can modify locally
  - Error-prone
- LDAP
  - Propagates real-time
  - Can’t be modified locally
  - Many limitations

Logging and alerting

- Email alerts
- All events to syslog
  - Make sure logs are centralized
  - Using syslog-ng sudo logs are automatically parsed and you can also push alerts to Slack, Splunk, Elasticsearch, etc.
- Debug logs
  - Debug rules
  - Report problems

sudo logs in Slack



my syslog-ng robot APP 3:08 PM

```
host: centos7.localdomain | program: sudo(14905) | severity: notice
czanik : TTY=pts/0 ; PWD=/home/czanik ; USER=root ; COMMAND=/bin/ls
Apr 23rd
host: centos7.localdomain | program: sudo(14916) | severity: notice
czanik : TTY=pts/0 ; PWD=/home/czanik ; USER=root ; COMMAND=/bin/ls /
Apr 23rd
host: centos7.localdomain | program: sudo(14922) | severity: notice
czanik : TTY=pts/0 ; PWD=/home/czanik ; USER=root ; COMMAND=/bin/ls /root
Apr 23rd
```

my syslog-ng robot APP 6:57 PM

```
host: centos7.localdomain | program: sudo(10462) | severity: notice
czanik : TTY=pts/0 ; PWD=/home/czanik ; USER=root ; COMMAND=/bin/ls /
Apr 23rd
host: centos7.localdomain | program: sudo(10642) | severity: notice
czanik : TTY=pts/0 ; PWD=/home/czanik ; USER=root ; COMMAND=/bin/joe
Apr 23rd
```

ONE IDENTITY by Quest

What is new sudo 1.9

- Recording Service: collect sudo IO logs centrally
- Audit Plugin
- Approval Plugin framework
- Python support for plugins
- Chroot and CWD

Recording Service

- Collect sudo IO logs centrally
- Streamed real-time, securely
- Convenient, available, secure

Audit plugin

- Not user visible
- API to access to all kinds of sudo logs

- Useful from Python
- Logging/Alerting to Elasticsearch, cloud providers, etc.
  - without external tools (like syslog-ng)

Approval Plugin framework

- Session approval
- No 3rd party binary plugin necessary
- A new policy without replacing the policy plugin

- Using Python, you can connect sudo with ticketing systems
  - Allow session only with open ticket
- Limit access to working days/hours

Python Support

- Extend sudo using Python
- Using the same API-s as C plugins

- API: https://www.sudo.ws/man/sudo_plugin.man.html
  - Python plugin documentation: https://www.sudo.ws/man/sudo_plugin_python.man.html

- No development environment or compilation is needed

Policy plugin API

- Decides who can do what
- Only one allowed
- Enabled in /etc/sudo.conf

- Example: only allow to run the command “id”

Policy plugin API example: code

```
import sudo
class SudoPolicyPlugin(sudo.Plugin):
    def check_policy(self, argv, env_add):
        cmd = argv[0]          # the first argument is the command name
        if cmd != "id":        # Example for a simple reject:
            sudo.log_error("You are not allowed to run this command!")
            return sudo.RC_REJECT
        command_info_out = ( # setup command to execute
            "command=/usr/bin/id",  # Absolute path of command
            "runas_uid=0",          # The user id
            "runas_gid=0")          # The group id
        return(sudo.RC_ACCEPT, command_info_out, argv, env_add)
```

ONE IDENTITY by Quest

Policy plugin API example: screenshot

```
[czanik@centos7 ~]$ sudo ls
You are not allowed to run this command!
[czanik@centos7 ~]$ sudo id
uid=0(root) gid=0(root) groups=0(root)
```

IO logs API

- Access input and output from user sessions
- Only one Python implementation is allowed
- Python examples:
  - Break connection if a given text appears on screen
  - Break connection if “rm -fr” is typed in the command line
  - Ask for the reason of the session

IO logs API example 1 (output check): code

```
import sudo

class MyIOPlugin(sudo.Plugin):
    def log_ttyout(self, buf):
        if "MySecret" in buf:
          sudo.log_info("Don't look at my secret!")
          return sudo.RC_REJECT
```

IO logs API example 1 (output check): screenshot

```
[czanik@centos7 ~]$ sudo -s
[root@centos7 czanik]# cd /root/
[root@centos7 ~]# ls
DoNotEnter  kick.py_v1  policy.py_v1  sng
kick.py     policy.py   __pycache__   sudo
[root@centos7 ~]# cd DoNotEnter/
[root@centos7 DoNotEnter]# ls
Don't look at my secret!
                           Hangup
[czanik@centos7 ~]$
```

IO logs API example 2 (input check): code

```
import sudo

class MyIOPlugin(sudo.Plugin):
    def __init__(self, version: str, plugin_options, **kwargs):
        self.collected_buf = ''

    def log_ttyout(self, buf):
        self.collected_buf += buf
        if "rm -fr" in self.collected_buf:
          sudo.log_info("Oops. 'rm -fr' is dangerous! Kicking you out...")
          return sudo.RC_REJECT
        # drop all the string until last enter:
        last_enter_pos = self.collected_buf.rfind("\n")
        if last_enter_pos >= 0:
          self.collected_buf = ''
```

IO logs API example 2 (input check): screenshot

```
[czanik@centos7 ~]$ sudo -s
[root@centos7 czanik]# ls
Desktop  Documents  Downloads  Music  Pictures  Public
Templates  Videos
[root@centos7 czanik]# rm -fOops. 'rm -fr' is dangerous!
Kicking you out...

                                        Hangup
```

IO logs API example 3 (conversation): code

```python
import sudo

class ReasonLoggerIOPlugin(sudo.Plugin):
    def open(self, argv, command_info):
        try:
            conv_timeout = 120  # in seconds
            sudo.log_info("Please provide your reason for executing '{}'".format(argv[0]))
            message1 = sudo.ConvMessage(sudo.CONV_PROMPT_ECHO_ON, "Reason: ", conv_timeout)
            message2 = sudo.ConvMessage(sudo.CONV_PROMPT_MASK, "Secret reason: ", conv_timeout)
            reply1, reply2 = sudo.conv(message1, message2)

            with open("/tmp/sudo_reasons.txt", "a") as file:
                print("Executed", ' '.join(argv), file=file)
                print("Reason:", reply1, file=file)
                print("Hidden reason:", reply2, file=file)

        except sudo.ConversationInterrupted:
            sudo.log_error("You did not answer in time")
            return sudo.RC_REJECT
```

IO logs API example 3 (conversation): screenshot

```
[czanik@centos7 ~]$ sudo -s
Please provide your reason for executing '/bin/bash'
Reason: my public reason
Secret reason: **************
[root@centos7 czanik]#
```

Group plugin API

- Allows non-Unix group lookups
- Example: can check if admin is on duty

- Python example: no password is used if user part of mygroup

```
Defaults group_plugin="python_plugin.so \
    ModulePath=/root/group.py \
    ClassName=SudoGroupPlugin"
%:mygroup ALL=(ALL) NOPASSWD: ALL
```

ONE IDENTITY by Quest

Group plugin API example

```
import sudo

class SudoGroupPlugin(sudo.Plugin):
    def query(self, user: str, group: str, user_pwd):
        hardcoded_user_groups = {
            "testgroup": [ "testuser1", "testuser2" ],
            "mygroup": [ "czanik" ]
        }
        group_has_user = user in hardcoded_user_groups.get(group, [])
        return sudo.RC_ACCEPT if group_has_user else sudo.RC_REJECT
```

ıd)

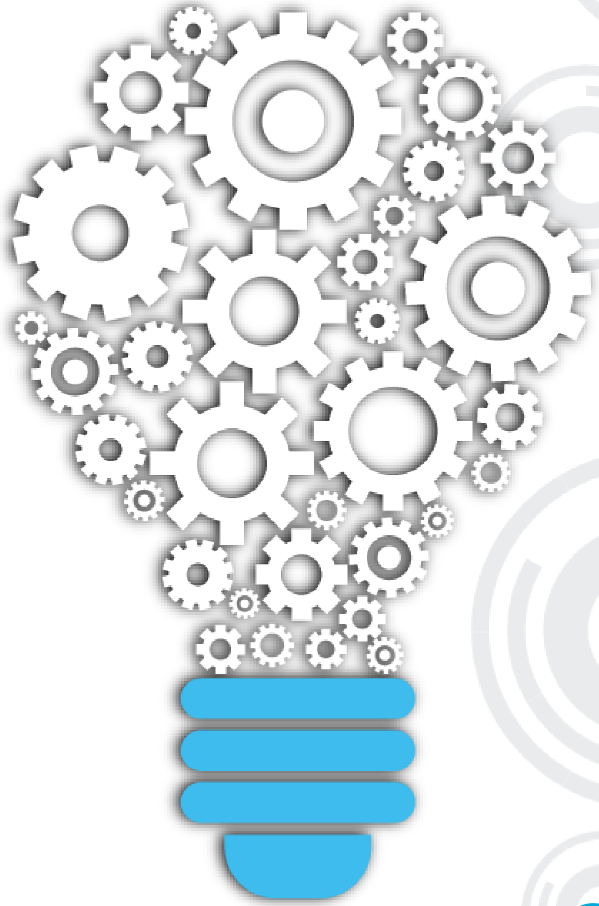
Chroot and CWD support

- Chroot access without full root access
- CWD support

- Must be enabled explicitly
- Fix directory or configurable by user

Not just a prefix, but…

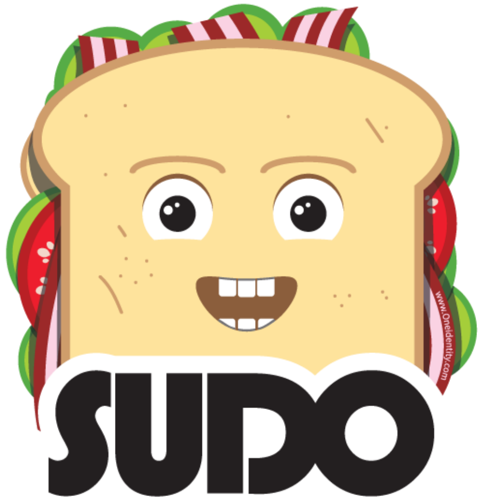
- 1.8
  - Fine-tuned permissions
  - Aliases / Defaults / Digest verification
  - Session recording / Logging and alerting
  - LDAP
  - Plugins
- 1.9
  - Python plugin
  - Logging API
  - Central session recording collection



ONE IDENTITY by Quest

Questions?

- Sudo website: https://www.sudo.ws/
- My email: peter.czanik@oneidentity.com
- Twitter: @Pczanik



ONE IDENTITY
by Quest

ONE IDENTITY

by Quest