What’s new in Sudo 1.8?

Pluggable modules for Sudo

Introduction to Sudo

- Sudo allows a system administrator to give users the ability to run commands as the super user without having to run a root shell, or su.
- Sudo logs each command run.
- Command to be run are prefixed with “sudo”.
- The file that determines who can run what is called the sudoers file.

Brief History of Sudo

- 1980: First version from SUNY/Buffalo
- 1985: Updated version posted to net.sources
- 1986: CU-Boulder version
  - Unix System Administration Handbook
- 1991: Root Group version
- 1994: I start making releases derived from the Root Group Sudo

Sudo 1.8 Modular Architecture

- Two types of plugins
  1. Policy plugins
     - Determine who can do what
     - Only one policy plugin may be loaded
  2. I/O log plugins
     - Record the session
       - tty input/output
       - stdin, stdout, stderr
     - More than one I/O log plugin may be loaded

What is this stuff good for?

- In the past, if an organization needed to use a different security policy for root access, users had to use the utilities provided with that policy.
  - I.E. replace sudo with something else
- Now, a policy plugin can be used instead
  - No longer need to stop using sudo
    - Your workflow doesn't have to change
    - Sudo behaves just like it always did

What is this stuff good for?

- Organizations may have a requirement to audit root access sessions
  - SOX, HIPPA, etc
- I/O logging since Sudo 1.7.3
  - Logged to a local file
    - Replay logs via sudoreplay
    - Syslog unsuitable due to large amount of traffic
  - If local logs are not enough, a plugin can be used

What if I like sudoers?

- Nothing changes
  - Sudoers (file or LDAP) is the default policy plugin
    - Visudo works the way it always has
    - No changes to logging, etc
  - Sudoers I/O log plugin logs to the local host, just like in Sudo 1.7
    - Sessions can be replayed with sudoreplay
    - Compressed with zlib by default (gzip)

Plugin API Design Decisions

Two main approaches to modules:

- Fine grained – lots of hooks
  - Logging, authentication, policy sources, etc
  - Pros
    - Easier to replace only certain parts of sudo
    - Simple modules might be smaller
  - Cons
    - Much more complicated API
    - harder to plug in a totally foreign policy server

Plugin API Design Decisions

- Coarse grained – module does most things
- Pros:
  - Simpler API
  - Architecture less limiting
- Cons:
  - modules may duplicate existing sudoers functionality

We have it both ways!

- The current plugin API is coarse-grained with a small number of entry points.
- Sudoers plugin already somewhat modular
  - Sudoers file vs. LDAP using nsswitch.conf
    - Could support pluggable sudoers sources
      - if there is demand for it…
  - Non-Unix group provider plugin API
    - Active Directory groups, alternate group file
    - Details later…

Who Benefits?

- Just our corporate overlords?
  - No!
- Several open source plugins in development
  - FreeIPA security server
  - University of Colorado extended LDAP plugin
  - I/O module for sslogger
  - I plan to work on a revamped sudoers format

/etc/sudo.conf

- Configures plugins and plugin-agnostic paths
- Plugin plugin_symbol plugin_path
  - `Plugin sudoers_policy sudoers.so`
  - `Plugin sudoers_io /usr/libexec/sudoers.so`
  - The “plugin_symbol” is the variable holding the `policy_plugin` or `io_plugin` struct
  - If “plugin_path” not fully qualified, it is relative to the libexec dir

/etc/sudo.conf

- Path name pathname
  - Currently only used to specify the askpass GUI prompter.
    - `Path askpass /usr/X11R6/bin/ssh-askpass`
    - `Path askpass /usr/libexec/ssh/gnome-ssh-askpass`
  - In Sudo 1.7 the askpass path was set in sudoers
  - Because user interaction is via the main sudo driver, not a plugin, askpass must be specified in sudo.conf

Sudo 1.8 Flow of Control

- Sudo parses command line options
- Reads /etc/sudo.conf
- Initializes plugins with user info and command line options
- Sudo queries plugin with command
  - User interaction via conversation function
  - Plugin returns yes, no, or error answer
  - Also sets up the execution environment

Sudo 1.8 Flow of Control

- Sudo runs the command
  - Changes UID/GID, CWD, environment, etc
  - If logging I/O
    - Command runs in a pty
    - I/O passed to I/O plugins for logging
      - I/O plugin can also terminate the command
  - When command exits, sudo calls plugin close() functions with the exit status (or error value)

Changes to Sudoers File

- askpass setting moved to sudo.conf
  - Sudoers module does not prompt user directly
- New iolog_file and iolog_dir settings
  - Where to put I/O logs and how to name them
  - Supports escape sequences that expand to user, group, runas_user, runas_group, hostname, command, as well as strftime() escapes.
- New group_plugin setting

Sudoers Group Provider Plugin

- Allows sudoers to support non-Unix groups
  - Can be useful to work around 16 group limit
- Simple API: init, query, cleanup functions
- Syntax:
  - Explicit: %:groupname
    - Only the group provider queried
  - Implicit: %groupname
    - Only queried if no Unix group by that name exists

Plugin API Walk Through

- Interface described in sudo_plugin.h header
- Plugin writer's guide bundled with Sudo 1.8 in POD and man formats. Html version at: http://www.sudo.ws/sudo_plugin.man.html
- Switch to shell to talk through sudo_plugin.h

Demo Time!

- Demo to show Sudo 1.7.5 vs. 1.8.0
  - Works the same
- Edit sudo.conf file to change plugin
- Run commands using custom plugin
- Show I/O logging and tty signal proxying
- Demo session replay using sudoreplay

Debunking the FUD

- A certain vendor has put out a “How Secure is Your Sudo” whitepaper advertised in Linux Magazine and others.
- Let’s debunk a few of their claims…

Debunking the FUD

- **Claim:** Sudo doesn't support remote logging or protect the integrity of the logs
- **Reality:** Sudo logs via syslog which can log remotely
  - A number of syslog daemons exist that use TLS and/or support encrypted, signed log files. Because sudo uses a standard log method, you can use a log daemon that suits your needs.
  - OS audit support on Linux, Solaris, MacOS, …

Debunking the FUD

- **Claim:** Sudo allows a user to run a shell, after which nothing is logged
- **Reality:** The intended use of sudo is to run commands without resorting to a root shell. The “noexec” feature can be used to disable shell escapes and “sudoedit” allows for safe editing of text files. When a shell is required, I/O logging can be used to record the session.

Debunking the FUD

- **Claim:** Lack of QA testing
- **Reality:** QA happens in several places
  - The amount of QA I do myself is increasing
    - Currently adding more regression and unit tests
  - Vendors that ship Sudo do QA of their packages
  - Open source means anyone can audit the code
  - Legitimate bugs get handled fairly quickly

Where To Get It?

- http://www.sudo.ws
- Source and binary packages available
- When third-party plugins are available I will maintain a list on the Sudo web site.