Sudo log scraping daemon and request form

Tim Ransom
tsranso@clemson.edu
Clemson University
Clemson, South Carolina

Abstract

Interactions of the users with the system itself can be an excellent source of knowledge regarding their requirements and desires. The sudo utility is a widely adopted permission authorization utility, included on the vast majority of Unix and Unix-like operating systems. In this work we extend the sudo application to include a front facing request form for application installation and scrape sudo logs to present information to the system administrators in a way that is easy to interpret, parse, and extend.

Keywords: Social and professional topics, professional topics, management of computing and information systems, System Management

ACM Reference Format:
Tim Ransom. 2020. Sudo log scraping daemon and request form. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 Introduction

System administration is a tedious and often difficult career within the information technology sector. Typical bureaucratic issues such as under-funding, understaffing, overextending are common sights for sysadmins, leading many to judiciously allocate their time to the most critical issues of the day. For this reason many sysadmins automate every process they can, gathering latent information into aggrigated forms for explicit review is a valuable use of time for a systems department. We have produced a system to procure, process, and present data regarding what users are trying to install (without permission).

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

2 Background

2.1 Privilege escalation applications

Over the years there have been many applications and systems to manage user privileges over a system. Some implementations of note include Yahoo's ssu application [9], userv from the GNU project [3] and recently doas from the OpenBSD community [4]. Sudo has become the standard application as it fits the vast majority of administration needs, is stable, and has been in circulation since 1980.

As all of these applications allow for users to cause damage to the system they all share a focus on the security of their code. Sudo has implemented a plugin system to allow users to extend to fit their use cases [2]

2.2 Linux logging systems

Linux has two logging systems in common use, often running concurrently. Journald is the modern logging option being a part of the systemd infrastructure. It typically stores it logs in a compressed binary format that is accessed by the journalctl utility. The syslog logging system is significantly older than journald and store it's logs as plaintext files which are then compressed once a significant amount has been accumulated.

Sudo makes heavy use of the syslog logging option, likely because it predates journald by several decades, is well integrated into standard C libraries, and is a well adopted convention. Logs output by syslog invocations are typically output to the `/var/log` directory, and modern sudo releases are printing specifically to `/var/log/auth` as specified in it's default configuration.

2.3 Sysadmin preferences

System administrators have a deep appreciation for command line applications. They consistently race higher than graphical interfaces in terms of trustworthiness, reliability, robustness, accuracy, and likability [8]. These results are shown in figure 2.3, and show a general trend of system administrators favoring command line. In particular we would like to draw attention to the robustness and reliability results. Being able to quickly rework existing tools to answer new questions is a hallmark of effective sysadmins, which is the direct application of robust/reliable applications.

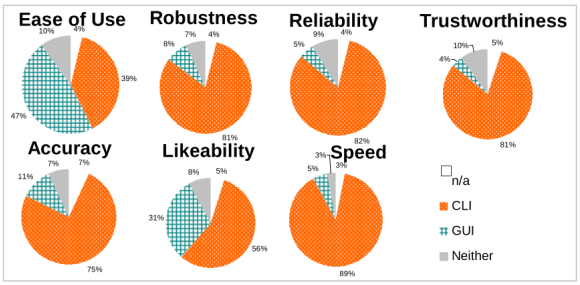

Figure 2. Comparative qualitative judgments of CLIs and GUIs

For similar reasons to command line interfaces, email is often a tool of choice for administrators. In addition to the direct line of communication to users it provides email is very commonly applied to produce system health reports generated by remote scripts. These plaintext messages can be processed with the same tools as any other text analyzer.

3 Development goals

3.1 Ease of Use and Extension

In order to be a genuinely useful tool, it must operate mostly independent of administrator action. There is a bounty of thoroughly tested and trusted applications to construct our application from, and the source code can be constructed from commonly understood tools should it need modified. Documentation is included as part of the literate programming style to aid an administrator understand the inner workings without an elder sage to help them.

3.2 Ease of Integration and Deployment

Scripts and daemons of all sizes include install scripts that place executable in known locations and verify library locations. We include one as well, which is integrated into the literate document described further in section 5.1.

4 Methodology and Design

4.1 Client server design

For our purposes the server is just any computer able to connect via ssh to all the machines to be monitored. The logs are pulled from the client machines over an ssh command and collated into a plaintext file for database storage. Security of the logs is relegated to the file permissions of ordinary Unix systems - the administrator may share the file if they choose but they must change the permissions themselves.

The clients here have a small utility copied into any directory in the global `$PATH` variable which we have named `socit-request-form`. This asks the user two questions: (1) if they would like to request some software to be installed and (2) if they have attempted to install this package locally. These questions are to establish that the user is aware of what they are doing and have at least thought about an alternative to use their requested software without needing assistance. A micro change to the sudoers plugin has also been made; such that when a failed sudo attempt is processed the system tells the user about the existence of the `socit-request-form`. We are attempting to illuminate the appropriate steps to installing or requesting software without opening these utilities to abuse.

4.2 A preference for standard tools

For the log gathering, there is a preference to use commonly installed tools from the Ubuntu repositories. It is not a far stretch to say that the utilities `ssh` and `grep` are installed, which are the only two tools (with some shell piping) required to gather the logs over the network. Once the report is generated there is one final call to `mail` to send the message to the `socit@clemson.edu` mailing list.

While scripting languages like shell can often be the best tool for collecting and organizing text, data processing is increasingly being done in Python. Note that all text processing could be done in a combination of awk and sed, but Python is currently a more used language and has powerful functional features such as filter and map.

The C programming language was chosen to implement the application install request form. This is not an application that users need to see the inner workings of and produces a small binary size to minimize network copy time and disk usage. The compilation and distribution of this executable is handled as part of the shell installation script.

5 Implementation

5.1 Literate style programming

Literate programming [6] has quitely been resurgent in computer science literature in the last few years [5]. Advantages of the literate style include integrated documentation in plain (not source code) language, inter-language interoperability, and critically for our application - reproducibility [7]. We make use of the same org mode emacs package described in the work by Schulte et al in order to leverage these advantages and neatly describe the process of constructing the project. (In fact this very document is part of that org mode file.)

5.2 Platform notes

This application is directly intended for the School of Computing at Clemson University. As such the target platform is the long term support version 18.04 of Ubuntu, this informs us to the location of logs, the content of log entries, and the supported build options for the application install request form. The sudo install on our School of Computing machines is generally standard, version 1.8.16, with the one notable exception of an additional PAM module to enable two factor authentication.

6 Results

The source code (in the lovely literate style) can be cloned from a git repository at https://github.com/ransomts/6240.

git, note that this does not include the backlog of Clemson user data. The output of the described system is quoted below, we present a section of the generated package install report that is emailed to the system administration and a sample interaction with the package install request form.

As with any software project, there should be a mention of the efficiency and possible bottlenecks in the developed systems. As the logs are accumulated over time on the server, the Python script to generate the report will eventually slow down. However this would require several orders of magnitude more logs than is reasonable to be generated by the users of the School of Computing hardware.

6.1 Display section of generated email

Below is an excerpt from the report generated by the Python script. Initial data was mined from the backlog of emails sent by sudo from 2018-2020. There are several classes of machines in use with different hardware stacks at Clemson. Being able to determine which packages are being requested for each class is a useful data set for staff to determine what to focus on.

```
total logs: 1244
requests for installs: 416

Most requested packages for all machines
Most requested logs, and how many times they
    were asked for:
['xz-utils', 9, ['software-properties-common', 9],
    ['build-essential', 10], ['atom', 10], ['zip', 12],
    ['unzip', 22]]

Most requested packages for machine class joey
Most requested logs, and how many times they
    were asked for:
['build-essential', 5, ['zip', 5], ['software-properties-
    common', 5], ['valgrind', 6], ['er.run', 6], ['un-
    zip', 15]]
```

6.2 Sudo error message addition

```
tsranso-ubuntu-vm 18:29 ~$ sudo ls /root
Password:
Sorry, try again.
Password:
Sorry, try again.
Password:
sudo: 3 incorrect password attempts, if you would
    like to request additional resources email
    socit@clemson.edu or run the command 'socit-
    request-form'
```

6.3 Display conversation session with user

```
tsranso-ubuntu-vm 17:20 ~$ socit-request-form
Would you like to submit a request to SoCIT to
    install this package? (yN) y
Before we begin, have you considered installing
    this package local to your home directory?
    (yN) y
Succinctly, why should this package be installed?
(Hitting enter sends the email, C-c to quit, you
    have one tweet of space (280 chars))

This package is super cool
email, has been sent :)
tsranso-ubuntu-vm 17:20 ~$
```

7 Conclusions

We have produced a small bit of source code to aid in a common task for system administrators - keeping up with the user requested packages to install. We note the current state of system administration as a combination of pipelining stable packages and leveraging modern python data processing to archive this. In addition there is a sizable amount of literature on the use of CLI interfaces in system administration contexts and literate programming to justify our choice in implementation. To end we would like to quote the fourth entry in the Unix philosophy presented by Doug McIlroy [1]: "Use tools in preference to unskilled help to lighten a programming task, even if you have to detour to build the tools and expect to throw some of them out after you've finished using them."

8 Bibliography

References

[1] [n.d.]. *BSTJ 57: 6. July-August 1978: UNIX Time-Sharing System: Forward. (McIlroy, M.D.; Pinson, E.N.; Tague, B.A.).* http://archive.org/details/bstj57-6-1899

[2] [n.d.]. *Sudo_SCALE9x.Pdf.* https://www.sudo.ws/slides/Sudo_SCALE9x.pdf

[3] [n.d.]. *Userv - User Services Client and Daemon.* https://www.gnu.org/software/userv/

[4] Nathan Holstein. [n.d.]. *Nholstein/OpenDoas.* https://github.com/nholstein/OpenDoas

[5] Mary Beth Kery, Marissa Radensky, Mahima Arya, Bonnie E. John, and Brad A. Myers. [n.d.]. The Story in the Notebook: Exploratory Data Science Using a Literate Programming Tool. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems* (Montreal QC, Canada, 2018-04-19) *(CHI '18)*. Association for Computing Machinery, 1–11. https://doi.org/10.1145/3173574.3173748

[6] D. E. Knuth. [n.d.]. Literate Programming. 27, 2 ([n. d.]), 97–111. https://doi.org/10.1093/comjnl/27.2.97

[7] Eric Schulte, Dan Davison, Thomas Dye, and Carsten Dominik. [n.d.]. A Multi-Language Computing Environment for Literate Programming and Reproducible Research. 46, 3 ([n. d.]). https://doi.org/10.18637/jss.v046.i03

[8] Leila Takayama and Eser Kandogan. [n.d.]. Trust as an Underlying Factor of System Administrator Interface Choice. In *CHI '06 Extended Abstracts on Human Factors in Computing Systems* (Montréal, Québec, Canada, 2006-04-21) *(CHI EA '06)*. Association for Computing Machinery, 1391–1396. https://doi.org/10.1145/1125451.1125708

[9] Christopher Thorpe. [n.d.]. SSU: Extending SSH for Secure Root Administration. ([n. d.]), 11.