


 [trimstray](#) / [the-practical-linux-hardening-guide](#) Public

This guide details creating a secure Linux production system. OpenSCAP (C2S/CIS, STIG).

 MIT license

 **10k** stars  **621** forks  Branches  Tags  Activity

 Star

 Notifications

 **Code**  Issues 7  Pull requests  Actions  Wiki  Security  Insights

master
1 Branch
0 Tags

...

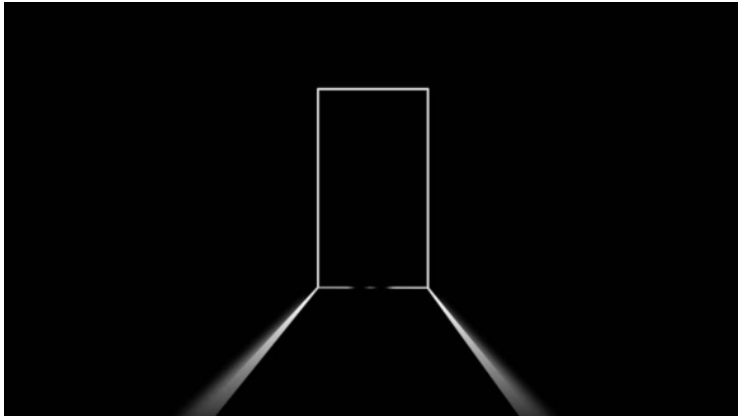
trimstray
uncommitted changes
✓

7b05a84 · 2 months ago

.github	update paths	6 years ago
docs	gh-page: updated url to wiki	6 years ago
lib	fixed typos; updated 'How to read this guide?'	6 years ago
static/img	minor updates	6 years ago
LICENSE.md	update license (MIT)	6 years ago
README.md	uncommitted changes	2 months ago

README
Code of conduct
MIT license

The Practical Linux Hardening Guide



"Did you know all your doors were locked?" - Riddick (The Chronicles of Riddick)

PRs
welcome
License
MIT

Table of Contents

- [Introduction](#)
 - [Prologue](#)
 - [The Importance of Hardening Linux](#)
 - [How to Harden Linux](#)
 - [Which Distribution Should be Used](#)
 - [How to Read This Guide](#)
 - [Okay. Let's start, 3, 2, 1... STOP!](#)
- [Policy Compliance](#)
 - [Center of Internet Security \(CIS\)](#)
 - [Security Technical Implementation Guide \(STIG\)](#)
 - [National Institute of Standards and Technology \(NIST\)](#)
 - [Payment Card Industry Data Security Standard \(PCI-DSS\)](#)
- [Security Content Automation Protocol \(SCAP\)](#)
 - [SCAP Security Guide](#)
 - [OpenSCAP Base](#)
 - [SCAP Workbench](#)
- [DevSec Hardening Framework](#)

- [Server Hardening Framework](#)
- [Contributing & Support](#)
- [License](#)

Introduction

Prologue

[The Practical Linux Hardening Guide](#) provides a high-level overview of hardening GNU/Linux systems. It is not an official standard or handbook but it *touches* and *uses* industry standards.

This guide also provides you with *practical step-by-step instructions* for building your own hardened systems and services. One of the main goals is to create a single document covering *internal* and *external* threats.

A few rules for this project:

- useful, simple, and not tiring
- include a lot of security tips from the C2S/CIS
- contains also non-related rules with C2S/CIS
- based on a minimal [RHEL7](#) and [CentOS 7](#) installations
- it's not exhaustive about Linux hardening
- some hardening rules/descriptions can be done better
- you can think of it as a checklist

The Practical Linux Hardening Guide use following [OpenSCAP](#) configurations:

- [U.S. Government Commercial Cloud Services \(C2S\) baseline inspired by CIS v2.1.1](#)
 - C2S for Red Hat Enterprise Linux 7 v0.1.43.
- [Red Hat Enterprise Linux 7 Security Technical Implementation Guide \(STIG\)](#)
 - The requirements are derived from the (NIST) 800-53 and related documents.

Please also remember:

■ This guide contains my comments that may differ from certain industry principles. If you are not sure what to do please see [Policy Compliance](#).

The Importance of Hardening Linux

Simply speaking, hardening is the process of making a system more secure. Out of the box, Linux servers don't come "hardened" (e.g. with the attack surface minimized). It's up to you to prepare for each eventuality and set up systems to notify you of any suspicious activity in the future.

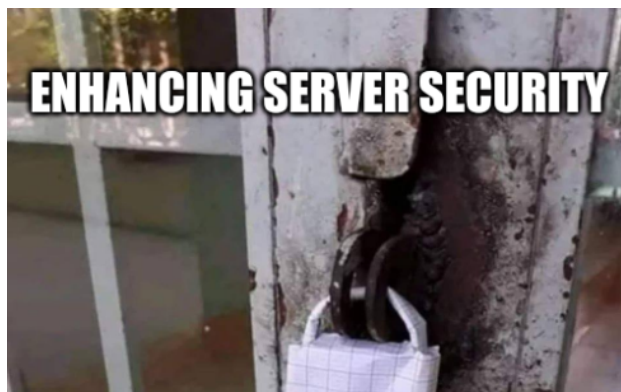
The process of hardening servers involves both IT ops. and security teams and require changes to the default configuration according to industry benchmarks.

Also for me, hardening is the fine art of doing the right things, even if they don't always look to have a big impact. It's always a balance between ease of use and protection.

You need to harden your system to protect your assets as much as possible. Why is it important? Please read a great, short article that [explains the hardening process](#) step by step by [Michael Boelen](#).

How to Harden Linux

In my opinion, you should drop all non-industry policies, articles, manuals, and others especially on production environments and standalone home servers. These lists exist to give a false sense of security and aren't based on authority standards.





There are a lot of great GNU/Linux hardening policies available to provide safer operating systems compatible with security protocols. For me, CIS and the STIGs compliances are about the best prescriptive guides - but of course you can choose a different one (e.g. PCI-DSS, DISA).

Most of all you should use [Security Benchmarks/Policies](#) which describe consensus best practices for the secure configuration of target systems.

Configuring your systems in compliance eliminates the most common vulnerabilities. For example, CIS has been shown to eliminate 80-95% of known vulnerabilities.

On the other hand, these standards are complicated checklists (often for newbies, difficult to implement). In my opinion, ideally, real world implementation is automated via something like OpenSCAP.

You should use a rational approach because more is not better. Each environment is different, so even though security rules should all work in theory, sometimes things will not work as expected.

Hardening is not a simple process. Here are general rules following common best practices:

- never use root account for anything that does not require it
- only `sudo` individual commands
- never set a server to run as root (except for initialization time) and ensure that it exits all unnecessary privileges before accepting requests
- secure your firewall the best you can and forbid all unnecessary access
- do not install unnecessary or unstable software

Which Distribution Should be Used

This guide is tested on **Red Hat Enterprise Linux 7** and **CentOS 7** distributions because these are:

- free (CentOS) and open source
- enterprise-class
- stable and reliable
- with great community support
- built on coherent snapshots of old packages

Both distributions allow the use of [certified tools](#) which can parse and evaluate each component of the SCAP standard.

If you use another distribution - no problem, this guide is also for you.

How to Read This Guide

Here is the structure of the chapters:

```
Chapter - e.g. Core Layer
|
|-- Subsection - e.g. Maintaining Software
|   \
|       |-- Rationale
|       |-- Solution (+ policies)
|       |-- Comments
|       |-- Useful resources
|
|-- Subsection - e.g. Accounts and Access
|   \
|       |-- Rationale
|       |-- Solution (+ policies)
|       |-- Comments
|       |-- Useful resources
```



Levels of understanding:

- *Chapter* and *subsection* offers a general overview
- *Rationale* tells you the reasoning behind the changes

- *solution* and *policies* are always compliant with the standard and on this basis, make changes
- *Comments* helps you figure out what you can change or add to the *solution*
- *Useful resources* provide deeper understanding

If you do not have the time to read hundreds of articles (just like me) this multipurpose handbook may be useful. This handbook does not get into all aspects of GNU/Linux hardening. I tried to put external resources in many places in this handbook in order to dispel any suspicion that may exist.

I did my best to make this guide a single and consistent (but now I know that is really hard). It's organized in an order that makes logical sense to me.

Do not treat this hardening guide as revealed knowledge. You should take a scientific approach when reading this document. If you have any doubts and disagree with me, please point out my mistakes. You should to discover cause and effect relationships by asking questions, carefully gathering and examining the evidence, and seeing if all the available information can be combined in to a logical answer.

I create this handbook for one more reason. Rather than starting from scratch in, I putting together a plan for answering your questions to help you find the best way to do things and ensure that you don't repeat my mistakes from the past.

So, what's most important:

- ask a questions about something that you observe
- do background research
- do tests with an experiments
- analyze and draw conclusions
- communicate results (for us!)

Okay. Let's start, 3, 2, 1... STOP!

Making major changes to your systems can be risky.

The most important rule of system hardening that reasonable admins follow is:

A production environment is the real instance of the app so make your changes on the dev/test!

The second most important rule is:

Don't do anything that will affect the availability of the service or your system.

The third rule is:

Make backups of the entire virtual machine and important components.

And the last rule is:

Think about what you actually do with your server.

Policy Compliance

Center of Internet Security (CIS)

The Center of Internet Security (CIS) is a nonprofit organization focused on improving public and private sector cybersecurity readiness and response.

Please see [CIS Benchmarks](#).

Security Technical Implementation Guide (STIG)

A Security Technical Implementation Guide (STIG) is a cybersecurity methodology for standardizing security protocols within networks, servers, computers, and logical designs to enhance overall security.

Please see [Stigviewer](#) to explore all stigs.

National Institute of Standards and Technology (NIST)

The National Institute of Standards and Technology (NIST) is a physical sciences laboratory and a non-regulatory agency of the United States Department of Commerce.

Please see [National Checklist Program \(NCP\)](#).

Payment Card Industry Data Security Standard (PCI-DSS)

Payment Card Industry Data Security Standard (PCI-DSS) compliance is a requirement for any business that stores, processes, or

transmits cardholder data.

In accordance with PCI-DSS requirements, establish a formal policy and supporting procedures for developing configuration standards for system components that are consistent with industry-accepted hardening standards like:

- Center of Internet Security (CIS)
- International Organization for Standardization (ISO)
- SysAdmin, Audit, Network, and Security (SANS) Institute
- National Institute of Standards and Technology (NIST)

Security Content Automation Protocol (SCAP)

Security Content Automation Protocol (SCAP) provides a mechanism to check configurations, vulnerability management and evaluate policy compliance for a variety of systems.

One of the most popular implementations of SCAP is OpenSCAP and it is very helpful for vulnerability assessment and as a hardening helper. OpenSCAP can easily handle the SCAP standards and generate neat, HTML-based reports.

Please see [SCAP Security Policies](#), [OpenSCAP User Manual](#), and [OpenSCAP Static](#).

SCAP Security Guide

The auditing system settings with SCAP Security Guide project contains guidance for settings for Red Hat/CentOS and it's validated by NIST.

You should inspect the security content of your system with `oscap info` module:

```
# For RHEL:
oscap info /usr/share/xml/scap/ssg/content/ssg-rhel7-ds.xml

# For CentOS:
oscap info /usr/share/xml/scap/ssg/content/ssg-centos7-ds.xml
```



OpenSCAP Base

The OpenSCAP scanner will only provide meaningful results if the content you want it to process is correct and up to date. The `oscap` tool scans your system, validates security compliance content, and generates reports and guides based on these scans.

Official [OpenSCAP Base](#) documentation says:

The command-line tool, called `oscap`, offers a multi-purpose tool designed to format content into documents or scan the system based on this content. Whether you want to evaluate DISA STIGs, NIST's USGCB, or Red Hat's Security Response Team's content, all are supported by OpenSCAP.

Before use, please read [Using OSCP](#) documentation.

```
# Installation:
yum install openscap-scanner

# Make a RHEL7 machine e.g. PCI-DSS compliant:
oscap xccdf eval --report report.html --profile xccdf_org.ssgproject.content_profile_pci-dss /usr/share/xml/scap/ssg/

# Make a CentOS machine e.g. PCI-DSS compliant:
oscap xccdf eval --report report.html --profile xccdf_org.ssgproject.content_profile_pci-dss /usr/share/xml/scap/ssg/
```



SCAP Workbench

SCAP Workbench is a utility that offers an easy way to perform common `oscap` tasks on local or remote systems.

Before use, please read [Using SCAP Workbench](#) documentation.

```
# Installation:
yum install scap-security-guide scap-workbench
```



DevSec Hardening Framework

Security + DevOps: Automatic Server Hardening.

This project covers some of the things in this guide which can be automated (e.g. setting of grub password or enforcing the permissions of the common directories). Its a good start if you want to make changes and see how it works from the level of

automation tools.

Project: [DevSec Hardening Framework](#) and  GitHub repository: [dev-sec](#).

Thanks [@artem-sidorenko](#)!

Contributing & Support

If you find something which doesn't make sense, or something doesn't seem right, please make a pull request and please add valid and well-reasoned explanations about your changes or comments.

Before adding a pull request, please see the [contributing guidelines](#).

If this project is useful and important for you or if you really like *The Practical Linux Hardening Guide*, you can bring **positive energy** by giving some **good words** or **supporting this project**. Thank you!

License

For license information, please see [LICENSE](#).

 To start please go to the [Wiki](#).

Releases

No releases published

Sponsor this project

 opencollective.com/trimstray

Packages

No packages published

Contributors 6

