NIST Special Publication
NIST SP 800-219r1 ipd

Automated Secure Configuration Guidance from the macOS Security Compliance Project (mSCP)

Initial Public Draft

Mark Trapnell
Eric Trapnell
Murugiah Souppaya
Bob Gendler
Karen Scarfone

This publication is available free of charge from:
https://doi.org/10.6028/NIST.SP.800-219r1.ipd

NIST NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY
U.S. DEPARTMENT OF COMMERCE

NIST Special Publication
NIST SP 800-219r1 ipd

Automated Secure Configuration Guidance from the macOS Security Compliance Project (mSCP)

Initial Public Draft

Mark Trapnell
Eric Trapnell
Murugiah Souppaya
Computer Security Division
Information Technology Laboratory

Bob Gendler
Customer Access and Support Division
Office of Information Systems Management

Karen Scarfone
Scarfone Cybersecurity

This publication is available free of charge from:
https://doi.org/10.6028/NIST.SP.800-219r1.ipd

March 2023

U.S. Department of Commerce
Gina M. Raimondo, Secretary

National Institute of Standards and Technology
Laurie E. Locascio, NIST Director and Undersecretary of Commerce for Standards and Technology

Certain commercial equipment, instruments, software, or materials, commercial or non-commercial, are identified in this paper in order to specify the experimental procedure adequately. Such identification does not imply recommendation or endorsement of any product or service by NIST, nor does it imply that the materials or equipment identified are necessarily the best available for the purpose.

There may be references in this publication to other publications currently under development by NIST in accordance with its assigned statutory responsibilities. The information in this publication, including concepts and methodologies, may be used by federal agencies even before the completion of such companion publications. Thus, until each publication is completed, current requirements, guidelines, and procedures, where they exist, remain operative. For planning and transition purposes, federal agencies may wish to closely follow the development of these new publications by NIST.

Organizations are encouraged to review all draft publications during public comment periods and provide feedback to NIST. Many NIST cybersecurity publications, other than the ones noted above, are available at https://csrc.nist.gov/publications.

Authority

This publication has been developed by NIST in accordance with its statutory responsibilities under the Federal Information Security Modernization Act (FISMA) of 2014, 44 U.S.C. § 3551 et seq., Public Law (P.L.) 113-283. NIST is responsible for developing information security standards and guidelines, including minimum requirements for federal information systems, but such standards and guidelines shall not apply to national security systems without the express approval of appropriate federal officials exercising policy authority over such systems. This guideline is consistent with the requirements of the Office of Management and Budget (OMB) Circular A-130.

Nothing in this publication should be taken to contradict the standards and guidelines made mandatory and binding on federal agencies by the Secretary of Commerce under statutory authority. Nor should these guidelines be interpreted as altering or superseding the existing authorities of the Secretary of Commerce, Director of the OMB, or any other federal official. This publication may be used by nongovernmental organizations on a voluntary basis and is not subject to copyright in the United States. Attribution would, however, be appreciated by NIST.

NIST Technical Series Policies

Copyright, Use, and Licensing Statements
NIST Technical Series Publication Identifier Syntax

Publication History

Approved by the NIST Editorial Review Board on YYYY-MM-DD [will be added upon final completion]
Supersedes NIST Series XXX (Month Year) DOI [will be added upon final completion]

How to Cite this NIST Technical Series Publication:

Trapnell M, Trapnell E, Souppaya MP, Gendler B, Scarfone K (2023) Automated Secure Configuration Guidance from the macOS Security Compliance Project (mSCP). (National Institute of Standards and Technology, Gaithersburg, MD), NIST Special Publication (SP) NIST SP 800-219r1 ipd. https://doi.org/10.6028/NIST.SP.800-219r1.ipd

Author ORCID iDs

Mark Trapnell: 0000-0002-5266-3610
Eric Trapnell: 0000-0001-9315-3732
Murugiah Souppaya: 0000-0002-8055-8527
Bob Gendler: 0000-0002-8928-6492

Karen Scarfone: 0000-0001-6334-9486

Public Comment Period

March 13, 2023 – April 27, 2023

Contact Information

applesec@nist.gov

National Institute of Standards and Technology
Attn: Computer Security Division, Information Technology Laboratory
100 Bureau Drive (Mail Stop 8930) Gaithersburg, MD 20899-8930

All comments are subject to release under the Freedom of Information Act (FOIA).

Abstract

The macOS Security Compliance Project (mSCP) provides resources that system administrators, security professionals, security policy authors, information security officers, and auditors can leverage to secure and assess macOS desktop and laptop system security in an automated way. This publication introduces the mSCP and gives an overview of the resources available from the project's GitHub site, which is continuously curated and updated to support each new release of macOS. The GitHub site provides practical, actionable recommendations in the form of secure baselines and associated rules. This publication also describes use cases for leveraging the mSCP content. Updates from the previous version of this publication mainly involve the new mSCP capability to create a custom benchmark by tailoring a baseline.

Keywords

Apple; baseline; configuration management; endpoint device security; macOS; macOS Security Compliance Project (mSCP); operating system security; security compliance.

Reports on Computer Systems Technology

The Information Technology Laboratory (ITL) at the National Institute of Standards and Technology (NIST) promotes the U.S. economy and public welfare by providing technical leadership for the Nation's measurement and standards infrastructure. ITL develops tests, test methods, reference data, proof of concept implementations, and technical analyses to advance the development and productive use of information technology. ITL's responsibilities include the development of management, administrative, technical, and physical standards and guidelines for the cost-effective security and privacy of other than national security-related information in federal information systems. The Special Publication 800-series reports on ITL's research, guidelines, and outreach efforts in information system security, and its collaborative activities with industry, government, and academic organizations.

Supplemental Content

The mSCP's GitHub site is at https://github.com/usnistgov/macos_security#readme, and the project documentation wiki is at https://github.com/usnistgov/macos_security/wiki.

Trademark Information

Call for Patent Claims

This public review includes a call for information on essential patent claims (claims whose use would be required for compliance with the guidance or requirements in this Information Technology Laboratory (ITL) draft publication). Such guidance and/or requirements may be directly stated in this ITL Publication or by reference to another publication. This call also includes disclosure, where known, of the existence of pending U.S. or foreign patent applications relating to this ITL draft publication and of any relevant unexpired U.S. or foreign patents.

ITL may require from the patent holder, or a party authorized to make assurances on its behalf, in written or electronic form, either:

a) assurance in the form of a general disclaimer to the effect that such party does not hold and does not currently intend holding any essential patent claim(s); or

b) assurance that a license to such essential patent claim(s) will be made available to applicants desiring to utilize the license for the purpose of complying with the guidance or requirements in this ITL draft publication either:

   i. under reasonable terms and conditions that are demonstrably free of any unfair discrimination; or

   ii. without compensation and under reasonable terms and conditions that are demonstrably free of any unfair discrimination.

Such assurance shall indicate that the patent holder (or third party authorized to make assurances on its behalf) will include in any documents transferring ownership of patents subject to the assurance, provisions sufficient to ensure that the commitments in the assurance are binding on the transferee, and that the transferee will similarly include appropriate provisions in the event of future transfers with the goal of binding each successor-in-interest.

The assurance shall also indicate that it is intended to be binding on successors-in-interest regardless of whether such provisions are included in the relevant transfer documents.

Such statements should be addressed to: applesec@nist.gov

Table of Contents

Table of Figures

193

Acknowledgments

The authors wish to thank Jason Blake, Blair Heiserman, and Stephanie Roberts from NIST; Allen Golbig from Jamf; Dan Brodjieski, Gary Gapinski, Elyse Anderson, and Joshua Glemza from NASA; and Jamie Richardson and Chris Stone from Apple for their contributions to the mSCP. The authors appreciate Bob McSulla and Ryan Jaynes from Tenable for developing audit files based on the mSCP, testing the baselines for different macOS versions, and contributing to Appendix C. The authors also thank Isabel Van Wyk from NIST for editing the document. Finally, portions of this document are based on content from the mSCP Wiki, so the work of all Wiki contributors is appreciated.

1. Introduction

The National Institute of Standards and Technology (NIST) has traditionally published secure configuration guides for Apple desktop/laptop operating system versions as prose-based Special Publications (SPs), such as NIST SP 800-179, Revision 1, *Guide to Securing Apple macOS 10.12 Systems for IT Professionals: A NIST Security Configuration Checklist*. In order to provide security configuration guidance to organizations more quickly and in a machine-consumable format, NIST established the open-source macOS Security Compliance Project (mSCP). NIST no longer produces SP guidance documents for each macOS release; instead, the mSCP continuously curates and updates machine-consumable macOS guidance. The latest macOS security baseline content is maintained and updated on the mSCP GitHub page [1].

Security baselines are groups of settings used to configure a system to meet a target level or set of requirements or to verify that a system complies with requirements. The mSCP seeks to simplify the macOS security development cycle by reducing the amount of effort required to implement security baselines. This collaboration between federal agencies minimizes duplicate effort that would otherwise be needed for these agencies to administer individual security baselines. Additionally, the secure baseline content provided is easily extensible by other parties to implement their own security requirements.

Organizations using mSCP content, particularly security baseline examples, should take a risk-based approach for selecting the appropriate settings and defining values that consider the context under which the baseline will be utilized.

1.1. Purpose and Scope

The purpose of this document is to introduce the mSCP to broader audiences. This document provides a high-level overview of the mSCP, its components, and some common use cases. It refers readers to the online project documentation for in-depth technical information and use instructions. This document is intended to be independent of macOS version releases; updates will be released as needed when there are substantial changes to the mSCP. Updates from the previous release of this document mainly involve the new mSCP capability to create a custom benchmark by tailoring a baseline.

The information in this document regarding the details of the mSCP GitHub site is accurate as of the time of publication. Readers seeking the latest detailed information on mSCP content or the content itself should visit the mSCP GitHub page and wiki.

Organizations that need to reference a NIST SP to demonstrate how they are complying with United States Government mandates for adopting secure configurations for their macOS devices may reference this SP instead of its deprecated predecessors, such as SP 800-179 or SP 800-179, Revision 1.

1.2. Audience

This document and the mSCP GitHub site are intended for system administrators, security professionals, policy authors, privacy officers, and auditors who have responsibilities involving

macOS security. Additionally, vendors of device management, security, configuration assessment, and compliance tools that support macOS may find this document and the GitHub site to be helpful.

1.3. Relevance to NIST SP 800-70 and the National Checklist Program

The security baselines from the mSCP GitHub page are included in the National Checklist Program. NIST SP 800-70, Revision 4 [2], explains that federal agencies are required to use appropriate security configuration checklists from the National Checklist Program when available. Part 39 of the Federal Acquisition Regulations, Section 39.101 paragraph (c) states,

> In acquiring information technology, agencies shall include the appropriate information technology security policies and requirements, including use of common security configurations available from the National Institute of Standards and Technology's website at https://checklists.nist.gov. Agency contracting officers should consult with the requiring official to ensure the appropriate standards are incorporated.

1.4. Document Structure

The remaining sections and appendices of this document are as follows:

- Section 2 provides an overview of the project, including what its goals are and how its content can be used.
- Section 3 explains the major components of the mSCP and provides pointers to additional information on component usage.
- The References section lists the references for the document.
- Appendix A briefly discusses how mSCP can help meet the needs of people in several roles.
- Appendix B provides examples of how a security professional might use mSCP content.
- Appendix C contains an example of how an assessment tool vendor could leverage mSCP content.
- Appendix D lists selected acronyms and abbreviations used in this document.

2. Project Description

The mSCP is an open-source project that provides a programmatic approach to generating and using macOS security configuration baselines. The project's content can be used to create customized security baselines of technical security controls by leveraging a library of rules, with each rule mapped to requirements in one or more existing security standards, regulations, or frameworks. This approach provides versioning and consistency of the content. Unifying and standardizing macOS baseline efforts via the mSCP means that updating security guidance is simplified and radically accelerated, even as new versions of macOS are introduced annually.

The mSCP started in August 2019 as a collaboration among operational IT security staff from NIST, the National Aeronautics and Space Administration (NASA), the Defense Information Systems Agency (DISA), and the Department of Energy's (DOE) Los Alamos National Laboratory (LANL).[1] The mSCP sought to map macOS settings to NIST SP 800-53, Revision 4 [3] with an extensible, modern approach to security guidance that could be used by any organization (e.g., government, enterprise, education) that needs to adhere to security compliance frameworks and policy.

As of this writing, the configuration settings represent guidance and best practices from NIST SP 800-53, Revision 5 [4]; NIST SP 800-171, Revision 2 [5]; the macOS DISA Security Technical Implementation Guide (STIG) [6]; the Committee on National Security Systems (CNSS) Instruction (CNSSI) Number 1253 [7]; the Center for Internet Security (CIS) Critical Security Controls Version 8 [8]; and internal organizational security guidance from NIST, NASA, and LANL.

2.1. Project Goals

Apple releases a new macOS version every year, and generally, agencies and organizations must wait for guidance or accept risk before deploying the new macOS version. Most agencies or organizations must create their own internal security configuration, which delays the deployment of the new macOS version or new hardware that only supports the new macOS version. The mSCP assists organizations in upgrading sooner. Generally, the technical security settings in macOS do not drastically change from release to release, with only a handful of new settings being introduced. By pursuing a rules-based approach, mSCP rules that remain applicable can be reused and incorporated into guidance for the latest macOS version. This enables quicker adoption of new security features that are not offered in prior versions of macOS.

The goals of the mSCP are to:

- Develop recommended security baselines using a risk-based approach
- Normalize and accelerate annual adoption of the new operating system and hardware by providing guidance to meet the security needs of new operating systems at the earliest availability
- Reduce worldwide efforts in creating annual guidance by unifying and consolidating compliance efforts into a single project

[1] See https://github.com/usnistgov/macos_security#authors for a current list of project contributors.

- Develop a methodology to foster collaboration between baseline authors, reducing overhead and redundancy
- Establish a unified approach for the configuration and assessment of controls across multiple sources and tools
- Enable the customization of existing content and the creation of new content, including creating custom baselines in order to meet organization-specific security requirements
- Provide device management and security tool vendors, auditors, and Apple insight into customer security configuration needs

2.2. mSCP Content Use

mSCP content can be used by any organization to assist in setting and assessing the security configuration of macOS systems. Security baselines can map to existing guidance or controls, such as those in NIST SP 800-53, Revision 5 [4], or they can be customized to meet an organization's specific needs. In mSCP terminology, a security baseline is represented as a *baseline file* that designates the rules for meeting a specific set of requirements. The mSCP provides a library of *rules* that are macOS settings. Each rule is mapped to a requirement within a security standard or framework. Baseline files and rules comprise much of the mSCP's content.

The mSCP offers several example baselines with descriptions adapted from Federal Information Processing Standards (FIPS) 199 [9], such as:

- The ***SP 800-53, Revision 5 low baseline*** is a defined map of controls to secure a system defined as a low-impact information system. The loss of confidentiality, integrity, or availability could be expected to have a **limited** adverse effect on organizational operations, organizational assets, or individuals.
- The ***SP 800-53, Revision 5 moderate baseline*** is a defined map of controls to secure a system defined as a moderate-impact information system. The loss of confidentiality, integrity, or availability could be expected to have a **serious** adverse effect on organizational operations, organizational assets, or individuals.
- The ***SP 800-53, Revision 5 high baseline*** is a defined map of controls to secure a system defined as a high-impact information system. The loss of confidentiality, integrity, or availability could be expected to have a **severe or catastrophic** adverse effect on organizational operations, organizational assets, or individuals.

Organizations using any baseline example should take a risk-based approach for selecting the appropriate settings and organizationally defined values depending on the context under which the baseline will be applied. Organizations can tailor any of the baselines to include controls specific to their needs and to produce evidence of control enforcement. Additional information on baseline customization can be found in SP 800-70 [2], which discusses the importance of customizing and testing baselines before applying them to a production system.

The mSCP provides scripts that can be used with baselines for several purposes, including the following:

- Creating scripts and profiles for configuring macOS

- Generating a mapping between security standards, regulations, frameworks, etc.
- Producing human-readable documentation in a variety of formats
- Customizing existing baselines

mSCP content can also be used to generate Security Content Automation Protocol (SCAP) content for automated security compliance scans. The SCAP generated follows the SCAP 1.3 specification [10]. The generation of SCAP content uses an Extensible Stylesheet Language Transformations (XSLT) file to create an Extensible Configuration Checklist Description Format (XCCDF) checklist document with an accompanying Open Vulnerability and Assessment Language (OVAL) document.

The XCCDF and OVAL documents are bundled into an SCAP data stream collection document with accompanying files that include Common Platform Enumeration (CPE) dictionary [11] information and an Open Checklist Interactive Language (OCIL) document. This creates an SCAP 1.3 document that validates using the NIST SCAP Content Validation Tool[2] and can be used by SCAP tools on macOS. More information on SCAP content generation is available at the project wiki.

[2] See https://csrc.nist.gov/projects/security-content-automation-protocol/scap-releases/scap-1-3.

3. mSCP Components

This section provides an overview of several components of the mSCP: security baseline files, configuration profiles and scripts, content generation scripts, customization capabilities, and directories. More information about all of these is available at the GitHub wiki.

3.1. Baselines and Benchmarks

The mSCP includes both baselines and benchmarks. A baseline is a catalog of recommended configuration settings, not a checklist or benchmark, and should be customized based on the organization's risk profile. Implementing every item is not likely to be possible or sensible in many operational scenarios. Baselines can be used to assist in the creation of security benchmarks. A *benchmark* differs from a baseline in that it defines values in addition to a set of controls. Benchmarks are published by organizations that have made risk-based decisions, such as DISA and CIS. Organizations can also define their own benchmark. These values are called Organization-Defined Values (ODVs), and they exist throughout the baselines and can be set during customization.

3.2. Security Baseline Files

In the mSCP, a security baseline is defined in a Yet Another Markup Language (YAML) file. A YAML file is a human-readable file format commonly used by configuration files where data are stored and/or transmitted. A baseline YAML file consists of the following required fields:

- **`title`** – A human-readable name for the baseline
- **`description`** – A short description of the baseline, including its use case and target operating system (OS) version
- **`authors`** – Developers of the baseline
- **`profile`** – The security content portion of the baseline
    - **`section`** – A keyword for organizing settings
    - **`rules`** – The names of the rule files that are a part of this baseline

The following code provides a partial example of a YAML file that illustrates the use of these fields (with field names bolded):

```
title: "Apple macOS 11 (Big Sur) Test Baseline"
description: |
  This guide describes the prudent actions to take when securing a macOS 11
system against the Test Baseline.
authors: |
  |===
  |Joe Doe|NIST
  |===
profile:
  - section: "Authentication"
    rules:
      - auth_pam_login_smartcard_enforce
      - auth_pam_su_smartcard_enforce
      - auth_pam_sudo_smartcard_enforce
      - auth_smartcard_allow
  - section: "Auditing"
    rules:
      - audit_acls_files_configure
      - audit_acls_files_mode_configure
      - audit_acls_folder_wheel_configure
```

Fig. 1. Security Baseline YAML File.

3.2.1. Rule File Composition

A YAML rule file is broken down into the following subsections. This list and the following example are from the Rules section of the mSCP wiki.

- **`id`** – The `id` should match the file name without the yaml file extension.
- **`title`** – The `title` is a human-readable title of the rule.
- **`discussion`** – The `discussion` should provide a concise description of the intended use of the rule.
- **`check`** – Every rule will have a `check`. A shell-based check should be able to validate and check most rules.
- **`result`** – Expected results from the check.
- **`fix`** – The `fix` will appear in a document when generated. If a `fix` includes `[source,bash]`, the `fix` will be used for generating the script to enforce the rule.
- **`references`** – The `references` include a Common Configuration Enumeration (CCE) identifier and a mapping of the security frameworks, guidance, and individual controls that have been mapped to the rule. See the official repository of NIST CCEs [12] for more information.
- **`macOS`** – The validated macOS version for this rule.
- **`odv`** – If a rule supports the ODV functionality, then the `odv` section should be present. At a minimum, this field should contain a `hint` (provides a description when tailoring a baseline) and a default value that replaces the `$ODV` variable.

- **tags** – `Tags` are keywords used to categorize and identify related rules, and they can be added to or modified as needed. `Tags` can also be used to make index-based searching of the rules faster and easier.
- **severity** – The severity level specified in the DISA STIG, if applicable.
- **mobileconfig** – The `mobileconfig` and `mobileconfig_info` subsections are related. If `mobileconfig` is set to `true`, the information required for creating the mobileconfig configuration profile is required in the `mobileconfig_info` area.

The following code provides a notional example of a YAML rule file (with field names bolded):

```
id: sysprefs_screensaver_timeout_enforce
title: "Enforce Screen Saver Timeout"
discussion: |
  The screen saver timeout _MUST_ be set to $ODV seconds or a shorter length
of time.

  This rule ensures that a full session lock is triggered within no more than
$ODV seconds of inactivity.
check: |
  /usr/bin/osascript -l JavaScript << EOS
  function run() {
    let timeout =
ObjC.unwrap($.NSUserDefaults.alloc.initWithSuiteName('com.apple.screensaver')
\
  .objectForKey('idleTime'))
    if ( timeout <= $ODV ) {
      return("true")
    } else {
      return("false")
    }
  }
  EOS
result:
  string: "true"
fix: |
  This is implemented by a Configuration Profile.
references:
  cce:
    - CCE-91074-5
  cci:
    - CCI-000057
  800-53r5:
    - AC-11
    - IA-11
  800-53r4:
    - AC-11
  srg:
    - SRG-OS-000029-GPOS-00010
  disa_stig:
    - APPL-12-000004
  800-171r2:
    - 3.1.10
  cis:
    benchmark:
      - 2.3.1 (level 1)
    controls v8:
      - 4.3
macOS:
  - "12.0"
```

```
odv:
  hint: "Number of seconds."
  default: 1200
  stig: 900
  cis_lvl1: 1200
  cis_lvl2: 1200
tags:
  - 800-53r5_moderate
  - 800-53r5_high
  - 800-53r5_low
  - 800-53r4_moderate
  - 800-53r4_high
  - 800-171
  - cnssi-1253
  - cis_lvl1
  - cis_lvl2
  - cisv8
  - stig
severity: "medium"
mobileconfig: true
mobileconfig_info:
  com.apple.screensaver:
    idleTime: $ODV
```

Fig. 2. YAML Rule File

3.2.2. Rule File Categories

The mSCP organizes YAML files in the `rules` directory into the following subdirectories, each of which corresponds to a category of settings:

- **`audit`** – OpenBSM[3]
- **`auth`** – Smartcard authentication
- **`icloud`** – Apple's iCloud/Apple ID service
- **`os`** – Rules to configure the operating system that do not fit into the other categories
- **`pwpolicy`** – Password policy
- **`system_settings`** or **`sysprefs`** – Settings controlled within the System Settings or System Preferences application

The `rules` directory also includes a **`supplemental`** subdirectory, which contains additional information that supports the guidance provided by the baselines. Supplemental content contains information for rules that are not part of an existing baseline but could be beneficial for certain use cases. Supplemental content may not have mappings and may not contain the YAML rule file check and fix sections mentioned in Section 3.2.1. Supplemental content can be added to enhance baselines where organizational requirements are different than the system baseline requirements.

[3] See OpenBSM at https://github.com/openbsm/openbsm.

3.3. Configuration Profiles and Scripts

When an mSCP YAML file is processed, it yields a configuration script and/or configuration profile (`mobileconfig` file) as outputs. Both are used to apply configuration settings to a system.

A *configuration profile* is an Extensible Markup Language (XML) formatted file with a `mobileconfig` extension that contains a configuration payload. macOS can automatically configure itself based on a `mobileconfig` file's contents upon execution. Configuration profiles offer a convenient, Apple-supported mechanism for applying security settings to a macOS environment. Additionally, they can be cryptographically signed to ensure integrity and authenticity. These factors make configuration profiles the preferred vehicle for configuration delivery. However, `mobileconfig` files cannot modify all macOS settings, so a configuration script is needed for those settings that are not supported. See the developer documentation page for an example configuration profile and brief descriptions of its properties.

A *configuration script* is a shell script that directly manipulates operating system files. The script content is derived from all YAML rule files that have a `mobileconfig` value of `false` and belong to the specified baseline. The YAML rule file must contain the `fix` section in order to generate its corresponding configuration script entry.

3.4. Content Generation Scripts

The mSCP provides several types of scripts for generating baselines, human-readable guidance, baseline compliance checkers, and other types of content. Each script is described below.

3.4.1. Generate Baseline Script

The `generate_baseline.py` Python script compiles a list of security rules into a single baseline YAML file. It can be used to modify an existing security baseline or create a new one. See the wiki for additional information.

3.4.2. Generate Guidance Script

The `generate_guidance.py` script can produce human-readable guidance and generate the macOS Security Compliance Tool, which is a Z shell script.

The `generate_guidance.py` script takes a baseline file and produces a human-readable guide with information from the YAML rules files. The script can create documentation in several formats but always generates an AsciiDoc file. AsciiDoc (`.adoc`) is a plain text format that uses markup conventions for traditional document formatting and organization. AsciiDoc files are easily transformable into many other formats via the `generate_guidance.py` script, including HTML, PDF, and Excel. The Excel format is particularly useful for quickly viewing all of the rules of a baseline, and it contains all of the data in the YAML rules files.

The `generate_guidance.py` script can also create configuration profiles (`mobileconfig` files) and the macOS Security Compliance Tool. Using the `-s` argument, the `generate_guidance.py` script will generate an `org.{baseline}.audit.plist` file and another script, the macOS Security Compliance Tool, that can check and remediate compliance settings. The `audit.plist`

file can be used to set an exemption to organizational rules for approved users so that compliance checks can succeed without findings. To create an exemption for a rule, the `exempt` field should be set to `true` and an `exempt_reason` should be added.

See the wiki for more information on the `generate_guidance.py` script.

3.4.3. macOS Security Compliance Tool

The `{baseline}_compliance.sh` script runs interactively by default. It can evaluate a system's conformance to a baseline or remediate any incorrectly configured settings. Alternatively, the script can autonomously assess a system with the `-check` argument or automatically remediate baseline settings with `-fix`.

The lines below provide an example of the results of running the script:

```
Thu Jan 21 15:09:41 UTC 2021 auth_pam_login_smartcard_enforce passed (Result:
2, Expected: {integer: 2})
Thu Jan 21 15:09:41 UTC 2021 auth_smartcard_allow passed (Result: 1,
Expected: {integer: 1})
Thu Jan 21 15:09:41 UTC 2021 auth_pam_sudo_smartcard_enforce passed (Result:
2, Expected: {integer: 2})
Thu Jan 21 15:09:41 UTC 2021
auth_smartcard_certificate_trust_enforce_moderate passed (Result: 2,
Expected: {integer: 2})
Thu Jan 21 15:09:41 UTC 2021 auth_smartcard_enforce has an exemption (Reason:
Broken Reader)
```

Fig. 3. Compliance Script Sample Output.

For more information on the macOS Security Compliance Tool script, see the wiki.

3.4.4. SCAP Generation Script

The SCAP generation script, `generate_scap.py` can generate an SCAP 1.3 document, XCCDF document, or OVAL file. The script builds content from available tags within the YAML files and does not need to be pointed to a baseline file.

For more information, see the wiki.

3.4.5. Generate Mapping Script

The `generate_mapping.py` script allows for the quick creation of custom rules and baselines for a compliance framework not published by the mSCP. The script requires a user-created comma-separated values (CSV) file containing control identifiers that maps to a new framework (CSV column 1) from another already defined by the project (CSV column 2). By default, the script is designed to map a framework to the NIST SP 800-53, Revision 5 [4] set of controls. Adding the `-f` argument allows for mapping to another supported framework. See the wiki for more information on the `generate_mapping.py` script.

3.5. Customization

Organizations should make the risk-based decision on what controls and rules to use and how to apply them, as stated by NIST SP 800-53, Revision 5 controls PL-10 and PL-11. Customization allows organizations to generate their own customized content outside of that provided by the project. Additionally, it allows them to add content for internal-only controls, which are not suitable for inclusion in a global baseline. Customization primarily takes place within the custom folder. Here are examples of customizations supported by mSCP:

- **Baselines:** A `baseline` folder can be included within the `custom` folder to create customized baselines that fit an organization's needs. These baseline files may include rule, section, and template customization (discussed below). An existing baseline can be configured to create a custom benchmark. Additionally, it is possible to customize an included benchmark, but in doing so, it may no longer be compliant with the original requirements of that benchmark.
- **Rules:** Existing rules can have their setting values overridden via the `custom` folder instead of modifying the mSCP-supplied rule file. New rules can be created and added to existing baselines or to user-defined baselines. Organizations can create their own discussions, checks, results, fixes, and mappings of rules to security frameworks not included in the project. In order to override an existing rule, the custom rule file name must match an existing rule so that the `generate_guidance.py` script will pick up the new values. New rules not included in mSCP must be listed in the baseline YAML file specified when running `generate_guidance.py`. Additional information on custom rules can be found in an article written by mSCP contributor Allen Golbig [13].
- **Sections:** Custom sections can be used to organize existing or custom YAML rule files. Sections defined in the `custom` folder must be included in a baseline YAML file in order to be used by `generate_guidance.py`.
- **Templates:** Custom templates can be used to define new template structures for the project and affect the organization and appearance of generated documentation. The template files must match the name of an existing template and will override that template when running `generate_guidance.py`.
- **Logos:** An organization can include a custom logo when running the `generate_guidance.py` script by using the `-l` argument to point to an image file.
- **Tailoring:** The `generate_baseline.py` script allows for a baseline to be tailored using the `-t` argument. During the tailoring process, the script will prompt for each control containing an ODV to have its values customized. If a value is not supplied to a control with an ODV, it will use the default value in the rule file. Refer to Appendix C for an example of tailoring with ODVs.

3.6. Directories

mSCP source code releases available on the mSCP GitHub include the following directories:

- **`baselines`** – Contains the defined YAML baseline files

- `build` – Holds scripts, documents, and configuration profiles generated by running scripts
- `custom` – Used for creating customized baselines, rules, sections, or templates to meet an organization's requirements
- `includes` – Contains the YAML-based libraries required for running the scripts
- `rules` – Contains YAML rule files with one rule per file
- `scripts` – Contains the content generation scripts and their required files
- `sections` – Defines the sections that correlate to the directories in the `rules` folder; each section has its own YAML file that contains the section name and description as it will appear in the generated guide, which is human-readable documentation
- `templates` – Includes AsciiDoc templates for generating an AsciiDoc guide

References

[1] macOS Security Compliance Project (2023) *macOS Security Compliance Project*. Available at https://github.com/usnistgov/macos_security

[2] Quinn SD, Souppaya MP, Cook MR, Scarfone KA (2018) National Checklist Program for IT Products: Guidelines for Checklist Users and Developers. (National Institute of Standards and Technology, Gaithersburg, MD), NIST Special Publication (SP) 800-70, Rev. 4. https://doi.org/10.6028/NIST.SP.800-70r4

[3] Joint Task Force Transformation Initiative (2013) Security and Privacy Controls for Federal Information Systems and Organizations. (National Institute of Standards and Technology, Gaithersburg, MD), NIST Special Publication (SP) 800-53, Rev. 4, Includes updates as of January 22, 2015. https://doi.org/10.6028/NIST.SP.800-53r4

[4] Joint Task Force (2020) Security and Privacy Controls for Information Systems and Organizations. (National Institute of Standards and Technology, Gaithersburg, MD), NIST Special Publication (SP) 800-53, Rev. 5. Includes updates as of December 10, 2020. https://doi.org/10.6028/NIST.SP.800-53r5

[5] Ross R, Pillitteri V, Dempsey K, Riddle M, Guissanie G (2020) Protecting Controlled Unclassified Information in Nonfederal Systems and Organizations. (National Institute of Standards and Technology, Gaithersburg, MD), NIST Special Publication (SP) 800-171, Rev. 2, Includes updates as of January 28, 2021. https://doi.org/10.6028/NIST.SP.800-171r2

[6] Department of Defense (2023) *DISA STIG for macOS*. Available at https://public.cyber.mil/stigs/downloads/?_dl_facet_stigs=mac-os

[7] Committee on National Security Systems (2014) Security Categorization and Control Selection for National Security Systems. (National Security Agency, Ft. Meade, MD), Committee on National Security Systems Instruction (CNSSI) No. 1253. Available at https://www.cnss.gov/CNSS/issuances/Instructions.cfm

[8] Center for Internet Security (2023) *CIS Critical Security Controls Version 8*. Available at https://www.cisecurity.org/controls/v8/

[9] National Institute of Standards and Technology (2004) Standards for Security Categorization of Federal Information and Information Systems. (U.S. Department of Commerce, Washington, DC), Federal Information Processing Standards Publication (FIPS) 199. https://doi.org/10.6028/NIST.FIPS.199

[10] Waltermire DA, Quinn SD, Booth H, III, Scarfone KA, Prisaca D (2018) The Technical Specification for the Security Content Automation Protocol (SCAP): SCAP Version 1.3. (National Institute of Standards and Technology, Gaithersburg, MD), NIST Special Publication (SP) 800-126, Rev. 3. https://doi.org/10.6028/NIST.SP.800-126r3

[11] National Institute of Standards and Technology (2023). *Official Common Platform Enumeration (CPE) dictionary*. Available at https://nvd.nist.gov/products/cpe

[12] National Institute of Standards and Technology (2022) *CCE Platform Listing*. Available at https://ncp.nist.gov/cce

[13] Golbig A (2021) *Getting to Know: macOS Security Compliance Project – Part 2*. Available at https://golbiga.medium.com/getting-to-know-macos-security-compliance-project-part-2-24131b60cdfb

Appendix A. mSCP User Roles

The mSCP was designed to meet the needs of different security roles. These perspectives are briefly examined below.

Security policy authors define the policies for their organizations. The customization and ease of extensibility offered by the mSCP facilitate new content creation. Policy authors will need to familiarize themselves with the YAML rule file format described in Section 3.2.1. Of particular interest is the ability to map rules directly to references. Additionally, the generate mapping script (Section 3.4.5) enhances portability between compliance frameworks.

System administrators and security professionals are responsible for configuring the systems under their purview. They implement the guidance issued by security policy authors. Security professionals may wish to generate baselines (Section 3.4.1), guidance (Section 3.4.2), and configuration using the macOS Security Compliance Tool (Section 3.4.3).

Auditors approach macOS security compliance from a validator perspective, seeking proof that a system is configured in the required way. They are more interested in system setting documentation and compliance evidence than technical tools, such as configuration scripts. Both of these needs can be met by mSCP tools. The generate guidance script (Section 3.4.2) provides the necessary documentation in a variety of formats, including HTML, PDF, and Excel. The macOS Security Compliance Tool (Section 3.4.3) assesses a system and produces a log of the results. Additionally, some auditors may be interested in examining YAML rule content directly (Section 3.2.1).

Information security officers have a variety of goals but are ultimately responsible for ensuring that systems are configured according to their organizational requirements. To accomplish this, they need policy documentation (Section 3.4.2) and the results of compliance scans (Section 3.4.3). Information security officers may also be responsible for reviewing the security rules proposed by the policy authors. If this is the case, they may be interested in YAML rule file components (Section 3.2.1).

Vendors of device management, security, configuration assessment, and compliance tools can produce a series of audit files based on mSCP content to support different macOS versions and associated security baselines. These audit files are maintained, tested, published, and supported by the tool vendors. Tool customers can download and import the content into the tool to assess the state of their system against a particular baseline in an automated way.

Specific audit files of the mSCP by tool vendors are described on the project wiki page. This content will be updated as contributing tool vendors develop new audit content.

Appendix B. Example of mSCP Usage by a Security Professional

This appendix provides examples of how a security professional might use mSCP content. People in other roles might perform some of the same actions. The examples illustrated below were accurate at the time of publication, but see the mSCP wiki for up-to-date usage guidance. Note that the mSCP scripts are not meant to replace enterprise-class configuration and management tools. Configurations should be tested on development systems before being deployed on end users' systems.

Preparing to use mSCP

All project components are available from the mSCP GitHub page [1] by navigating to `Releases` and downloading the latest source code revision for the desired macOS version. Alternatively, the project source code can be downloaded via git, as the example below illustrates.

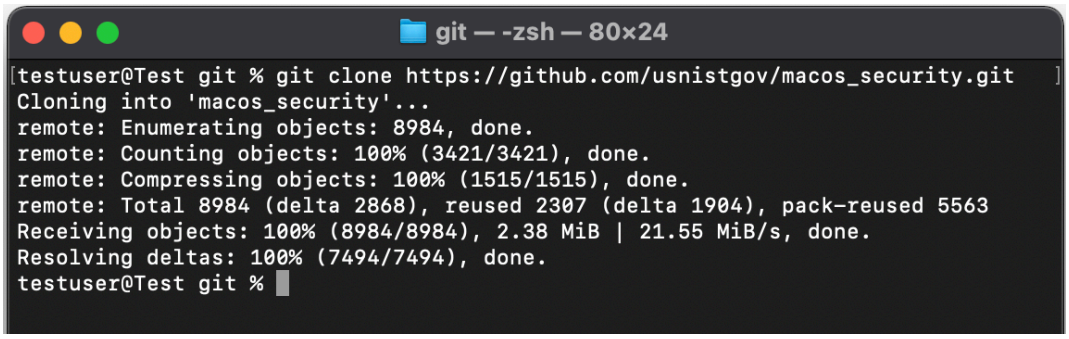


Fig. 4. Downloading the mSCP code.

mSCP components rely on prerequisite software listed on the Getting Started page, and any missing software must be installed.

Changing code branches and generating a baseline

After obtaining a copy of the source code, change the directory to the mSCP git folder, `macos_security`.

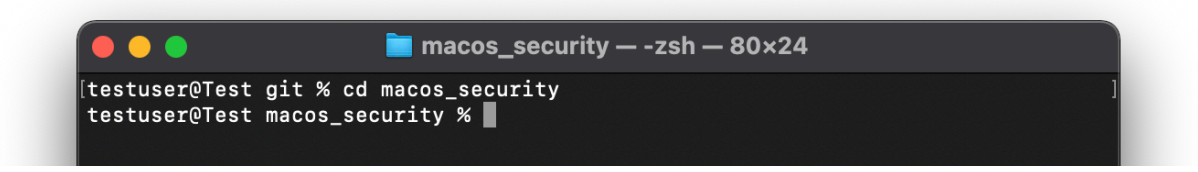


Fig. 5. Changing the directory to the mSCP git folder.

Next, select the appropriate code branch that corresponds to the target OS version. Then choose a baseline and use the `generate_baseline.py` script to create a baseline YAML file. The

example below illustrates these steps for the NIST SP 800-53, Revision 5 moderate baseline for macOS Big Sur.

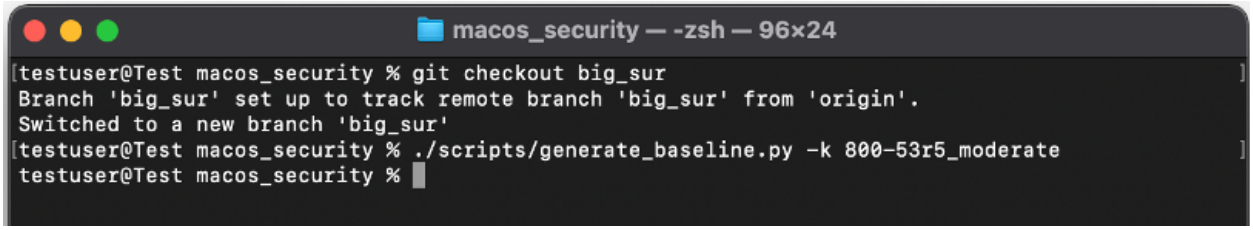


Fig. 6. Changing code branches and generating a baseline.

Creating the macOS Security Compliance Tool and configuration profiles

Using the `generate_guidance.py` script, create the macOS Security Compliance Tool and configuration profiles. The example below illustrates this, continuing from the previous example.

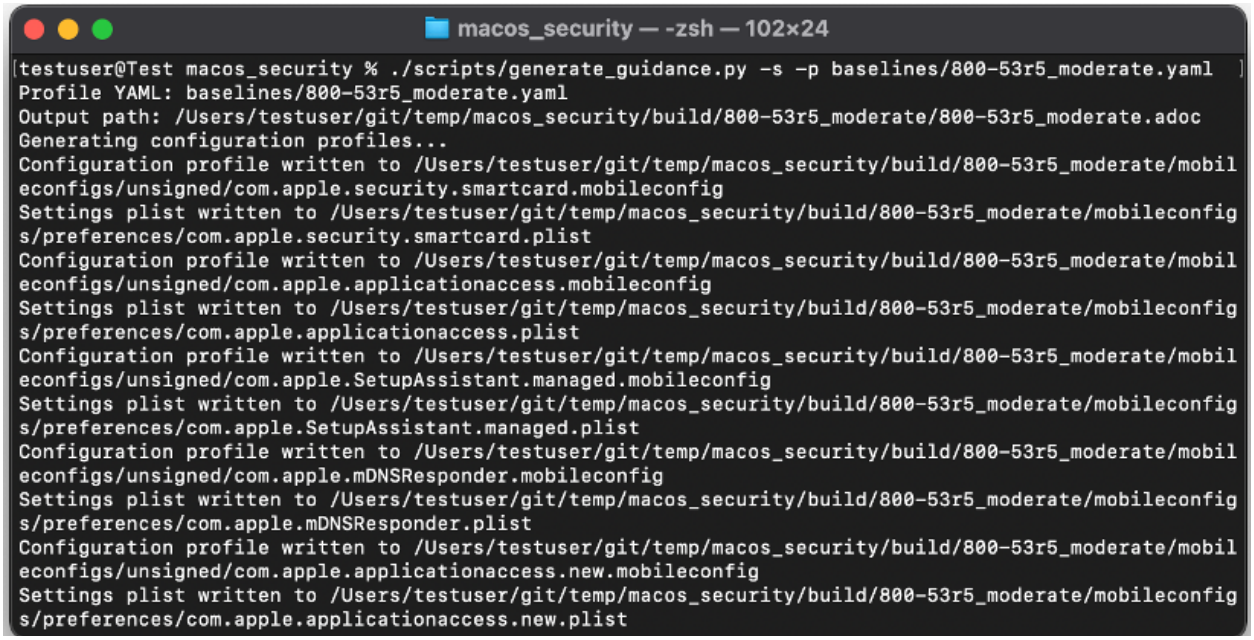


Fig. 7. Generating the compliance checker script and configuration profiles.

Running a compliance scan

As the example below shows, the macOS Security Compliance Tool is typically run with administrator privileges so that it can access all of the settings.

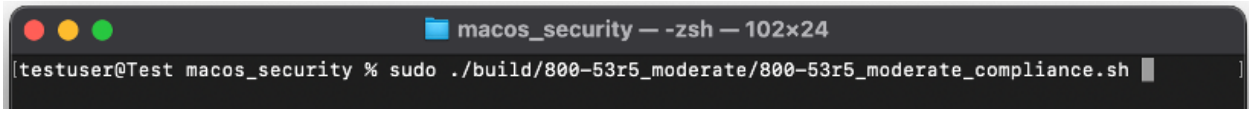


Fig. 8. Running the compliance checker script.

The example below shows the main menu presented by the macOS Security Compliance Tool.

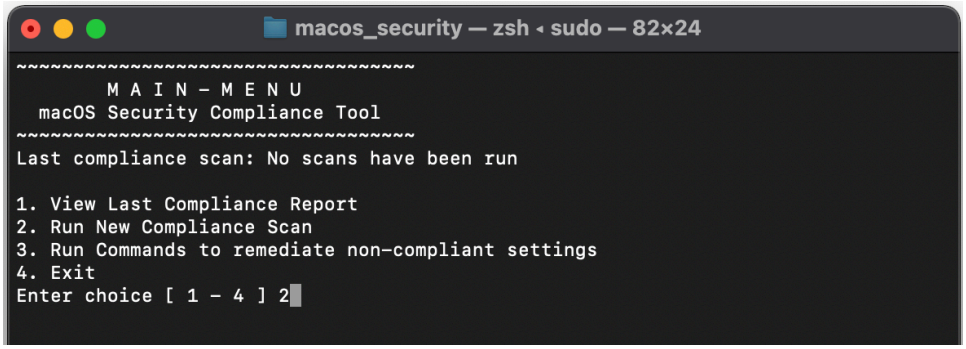


Fig. 9. Selecting “Run New Compliance Scan” from the main menu.

Selecting option 2, “Run New Compliance Scan,” from the main menu launches the scan. The example below shows output from the scan, which in this case reflects numerous rule failures, each indicating a deviation from the expected configuration.

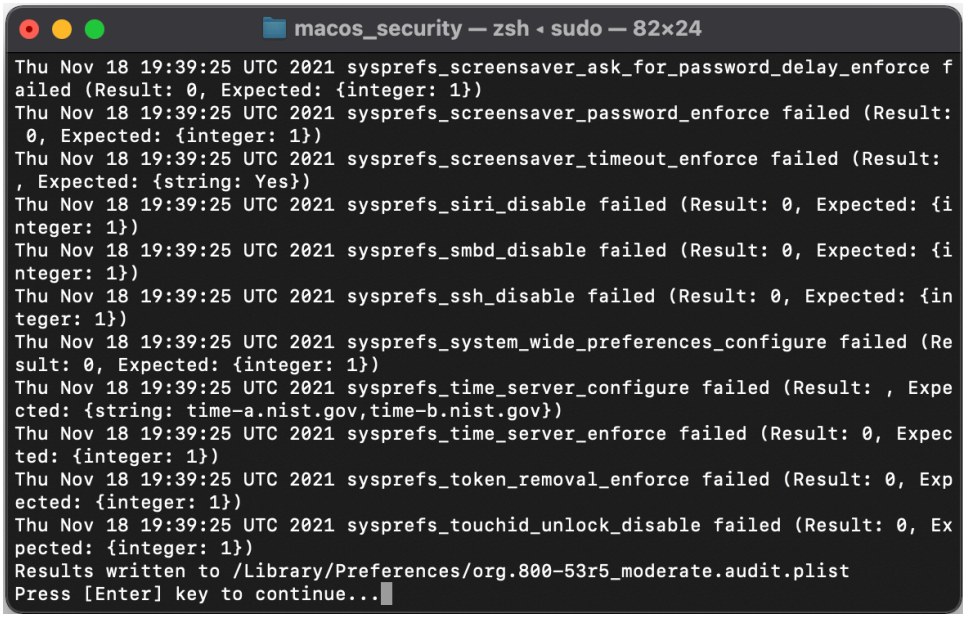


Fig. 10. Compliance scan output.

Selecting option 1, "View Last Compliance Report," from the main menu displays a summary of the compliance report results. The example below depicts results indicating that 30 tests passed and 108 tests failed for an overall score of 21.74 % compliant.

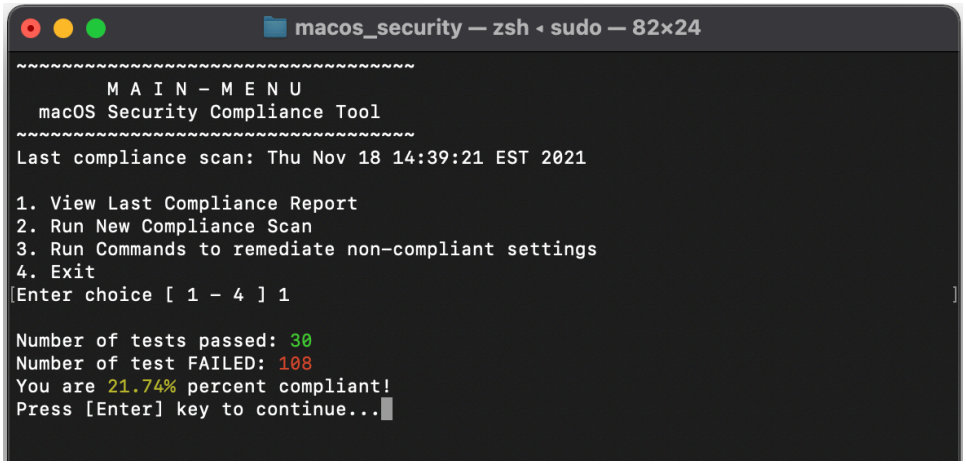


Fig. 11. Viewing a compliance report.

Fixing non-compliant settings

Selecting option 3, "Run Commands to remediate non-compliant settings," begins the process of fixing non-compliant settings discovered during a previous compliance scan. The example below illustrates the disclaimer to be reviewed and accepted before fixes are initiated. This disclaimer indicates the potential risk in applying fixes.

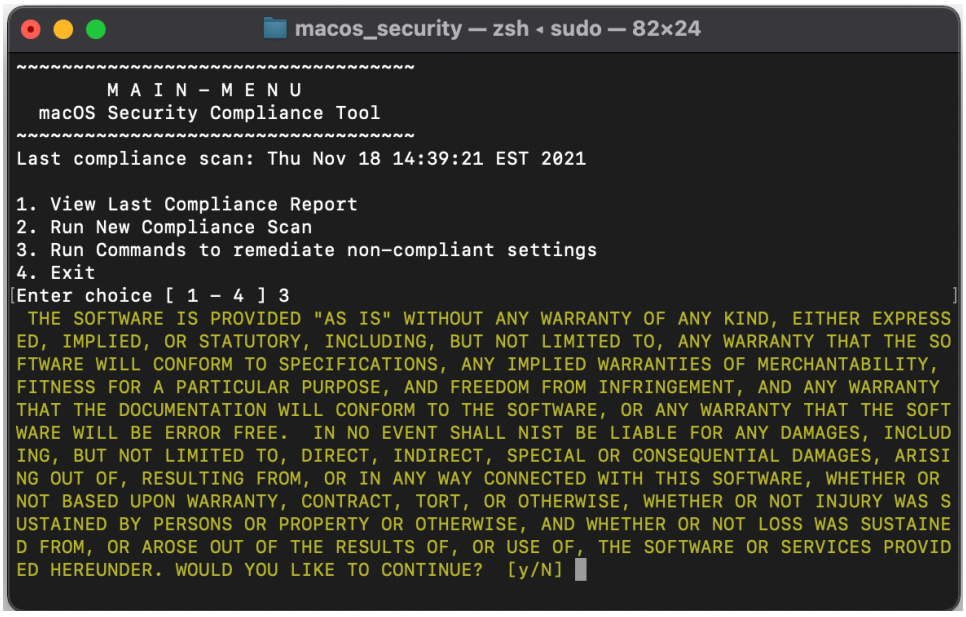


Fig. 12. Disclaimer for non-compliant settings remediation.

After the disclaimer statement is accepted, the fixes are applied to the system, as the example below illustrates.

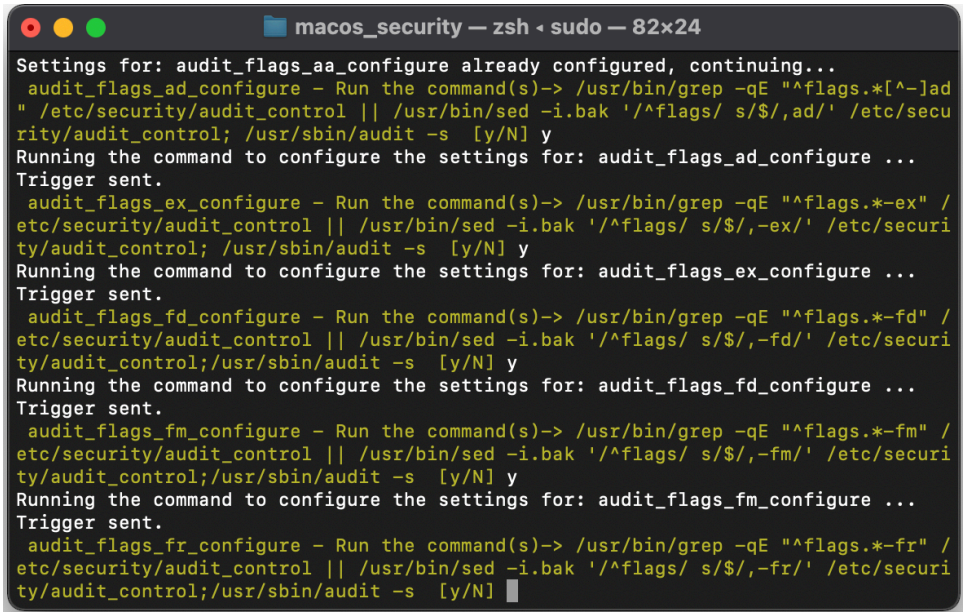


Fig. 13. Interactively configuring settings.

Appendix C. Example of Creating a Benchmark Using ODVs

This appendix provides an example of tailoring a baseline to create a custom benchmark using the `generate_baseline.py` script. The screenshot below illustrates the first step to creating a benchmark.

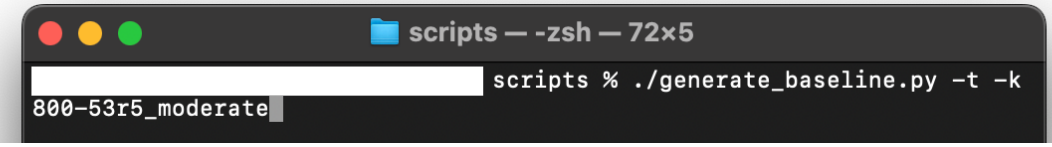


Fig. 14. Generate baseline command.

The `-t` option for `generate_baseline.py` is used to customize the specified baseline. The script prompts for a name for the benchmark being created, as the example below shows.

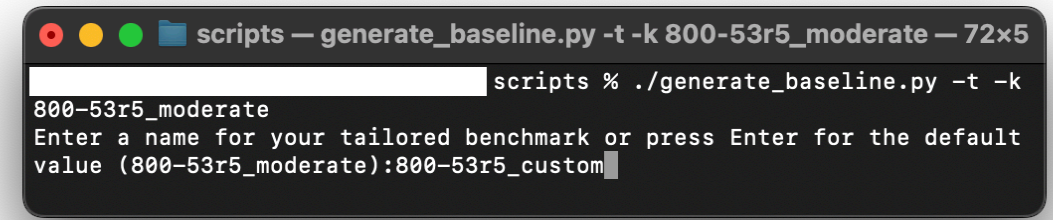


Fig. 15. Prompt for benchmark name.

Next, for each rule that exists in the specified starting baseline, the script asks if it should be included in the custom benchmark. An example is shown below.

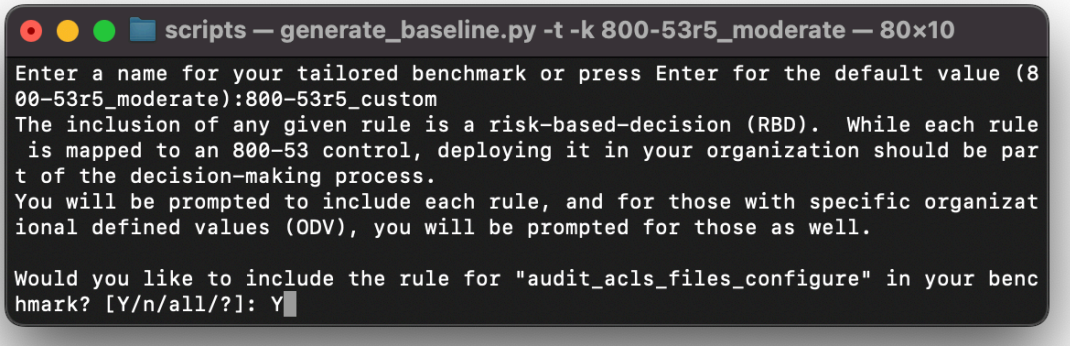
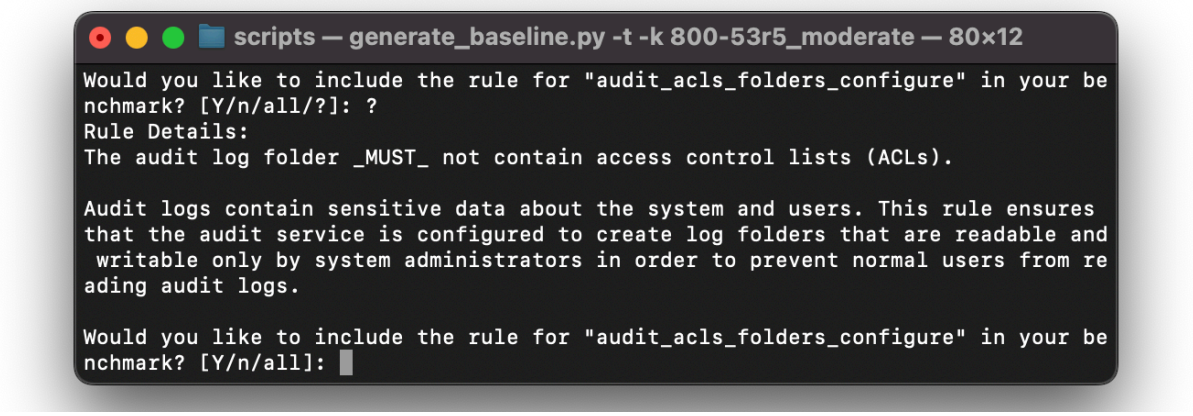


Fig. 16. Prompt for rule file inclusion.

816 Entering a "?" in response to a rule being included will display a description of that rule, as
817 illustrated below.

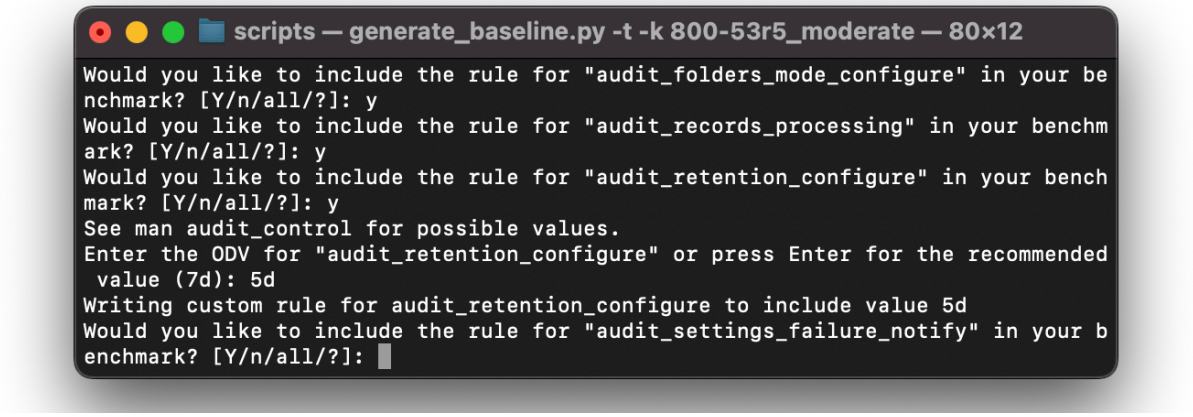


818

819 Fig. 17. Display rule description.

820

821 If a rule accepts an ODV, the script asks the user to enter their own value or use the default value
822 displayed. The example below illustrates this.



823

824 Fig. 18. Rule prompting for an ODV.

825

Appendix D. Example of mSCP Usage by an Assessment Tool Vendor

This appendix provides an example of how an assessment tool vendor converted mSCP content to their tool's proprietary format so that their tool could perform compliance checks against mSCP baselines and rules. Refer to the mSCP GitHub wiki page for the most current list of tool vendors and associated content that will support the mSCP baselines.

This example is for Tenable, Inc. They automated the conversion of mSCP YAML rules into their `.audit` format using Python and YAML libraries. Programmatically approaching this conversion allows for faster future releases and greater consistency, and it also maintains the integrity of the source content. Because the YAML content is all command-driven, it is converted to Tenable's `CMD_EXEC` check type for use with the Unix plugin. The YAML rules have a "tags" section that was used to create unique audit profiles related to common frameworks. An example of these profiles can be seen in the audit file naming convention:

- `NIST_macOS_Big_Sur_800-171_v1.4.0.audit`
- `NIST_macOS_Catalina_800-53r5_high_v1.5.0.audit`

See Tenable's research highlight for more details.

The following example shows a YAML-to-audit-check conversion. The content has been condensed and abbreviated for the purposes of comparison:

mSCP YAML

```
title: "Limit SSHD to FIPS 140 Validated Ciphers"

discussion: |
  If SSHD is enabled, then it _MUST_ be configured to limit the ciphers to
algorithms that are FIPS 140-validated.
FIPS 140-2 is the current standard for validating that mechanisms used to
access cryptographic modules utilize authentication that meets federal
requirements.
Operating systems utilizing encryption _MUST_ use FIPS-validated mechanisms
for authenticating to cryptographic modules.
  NOTE: /etc/ssh/sshd_config will be automatically modified to its original
state following any update or major upgrade to the operating system.

check: |
  /usr/bin/grep -c "^Ciphers aes256-ctr,aes192-ctr,aes128-ctr"
/etc/ssh/sshd_config

result:
  integer: 1
```

Tenable Audit Check

```
<custom_item>
  system      : "Darwin"
  type        : CMD_EXEC
  description : "Big Sur - Limit SSHD to FIPS 140 Validated Ciphers"
  info        : "If SSHD is enabled, then it _MUST_ be configured to limit
the ciphers to algorithms that are FIPS 140-validated.
FIPS 140-2 is the current standard for validating that mechanisms used to
access cryptographic modules utilize authentication that meets federal
requirements.
```

```
874   Operating systems utilizing encryption _MUST_ use FIPS-validated mechanisms
875   for authenticating to cryptographic modules.
876   NOTE: /etc/ssh/sshd_config will be automatically modified to its original
877   state following any update or major upgrade to the operating system."
878     cmd         : "/usr/bin/grep -c \"^Ciphers aes256-ctr,aes192-ctr,aes128-
879   ctr\" /etc/ssh/sshd_config"
880     expect      : "1"
881   </custom_item>
```

Appendix E. List of Symbols, Abbreviations, and Acronyms

Selected acronyms and abbreviations used in this paper are defined below.

CCE
Common Configuration Enumeration

CIS
Center for Internet Security

CNSS
Committee on National Security Systems

CNSSI
Committee on National Security Systems Instruction

DISA
Defense Information Systems Agency

FIPS
Federal Information Processing Standards

LANL
Los Alamos National Laboratory

mSCP
macOS Security Compliance Project

NASA
National Aeronautics and Space Administration

NIST
National Institute of Standards and Technology

ODV
Organization-Defined Value

OVAL
Open Vulnerability and Assessment Language

SCAP
Security Content Automation Protocol

SP
Special Publication

STIG
Security Technical Implementation Guide

XCCDF
Extensible Configuration Checklist Description Format

YAML
Yet Another Markup Language

Appendix F. Change Log

In March 2023, the following changes were made to this report:

- Made minor editorial changes throughout the report to improve clarity and usability
- Reformatted all content and revised front matter to follow the latest NIST technical report template
- Merged the content of the Executive Summary into Section 1 and deleted the Executive Summary section
- Section 1.1 – Summarized updates from the previous release
- Section 3.1 – Added a new subsection to define and discuss baselines and benchmarks
- Section 3.2.1 – Updated descriptions to match the project wiki and changed the example rule file to one with an ODV
- Section 3.4.4 – Changed from OVAL generation script to SCAP generation script
- Section 3.5 – Added discussion on tailoring baselines and benchmarks using the generate_baseline.py script and ODVs
- Appendix C – Created a new appendix showing an example of how to tailor a baseline to create a custom benchmark