Forward-Looking Statements

This presentation may contain forward-looking statements regarding future events, plans or the expected financial performance of our company, including our expectations regarding our products, technology, strategy, customers, markets, acquisitions and investments. These statements reflect management's current expectations, estimates and assumptions based on the information currently available to us. These forward-looking statements are not guarantees of future performance and involve significant risks, uncertainties and other factors that may cause our actual results, performance or achievements to be materially different from results, performance or achievements expressed or implied by the forward-looking statements contained in this presentation.

For additional information about factors that could cause actual results to differ materially from those described in the forward-looking statements made in this presentation, please refer to our periodic reports and other filings with the SEC, including the risk factors identified in our most recent quarterly reports on Form 10-Q and annual reports on Form 10-K, copies of which may be obtained by visiting the Splunk Investor Relations website at www.investors.splunk.com or the SEC's website at www.sec.gov. The forward-looking statements made in this presentation are made as of the time and date of this presentation. If reviewed after the initial presentation, even if made available by us, on our website or otherwise, it may not contain current or accurate information. We disclaim any obligation to update or revise any forward-looking statement based on new information, future events or otherwise, except as required by applicable law.

In addition, any information about our roadmap outlines our general product direction and is subject to change at any time without notice. It is for informational purposes only and shall not be incorporated into any contract or other commitment. We undertake no obligation either to develop the features or functionalities described, in beta or in preview (used interchangeably), or to include any such feature or functionality in a future release.

splunk> .conf22

Forget Add-on Builder! Build Splunk Platform Apps with Config Explorer!

DEV1160B

Brett Adams
Cyber Analytics Specialist | Deloitte

Brett Adams

Cyber Analytics Specialist | Deloitte

splunk> .conf22

Why not Splunk® Add-on Builder?

Is a great starting point for small projects and first time developers.

Has limited checkpointing capabilities.

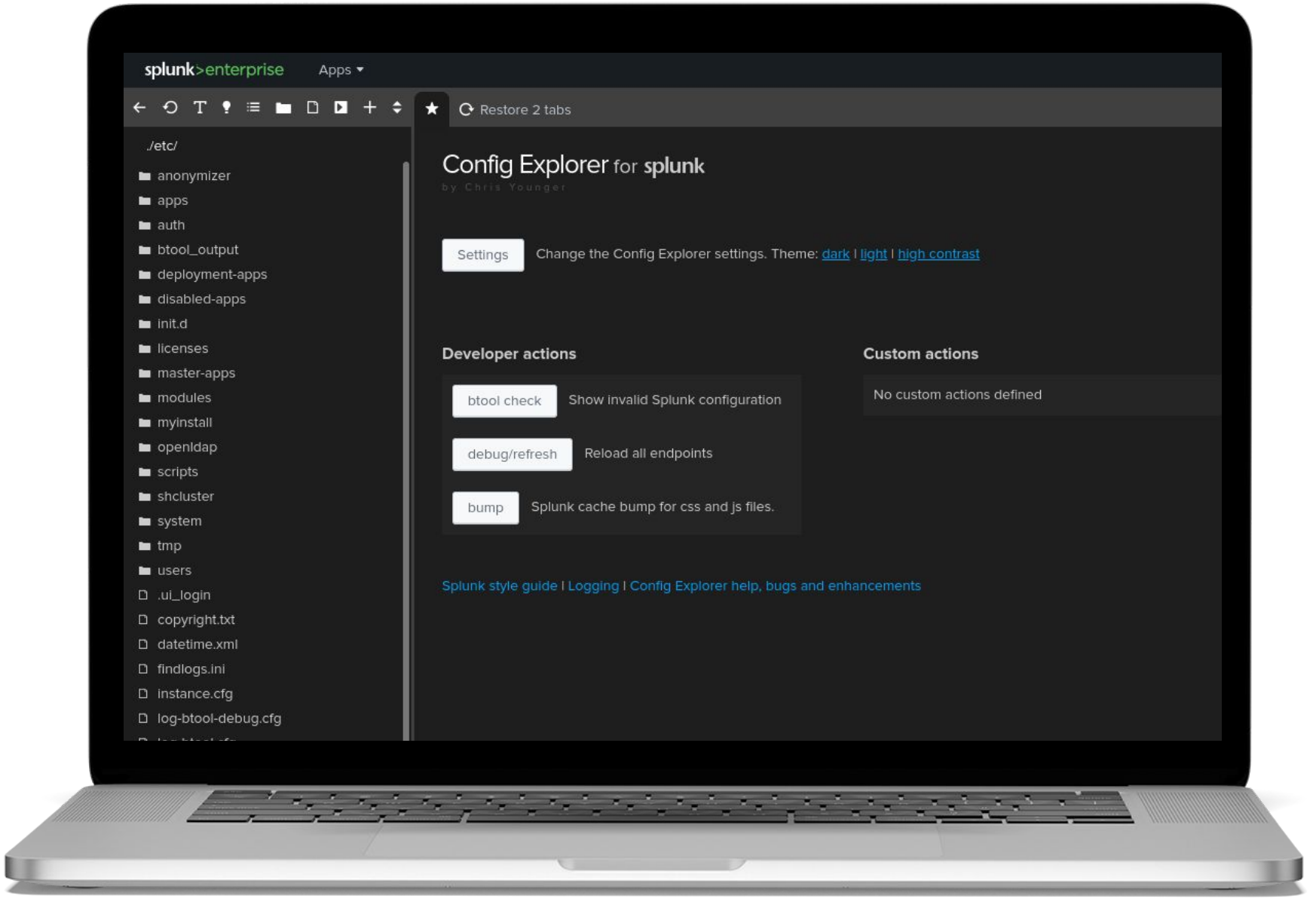
Creates large applications with multiple python dependencies.

By not using Add-on builder, you learn how Splunk apps actually work.

Config Explorer

Built by Chris Younger

Provides an editor interface for viewing and editing Splunk files.



Chris Younger

The Man, The Myth, The Splunkbase Legend

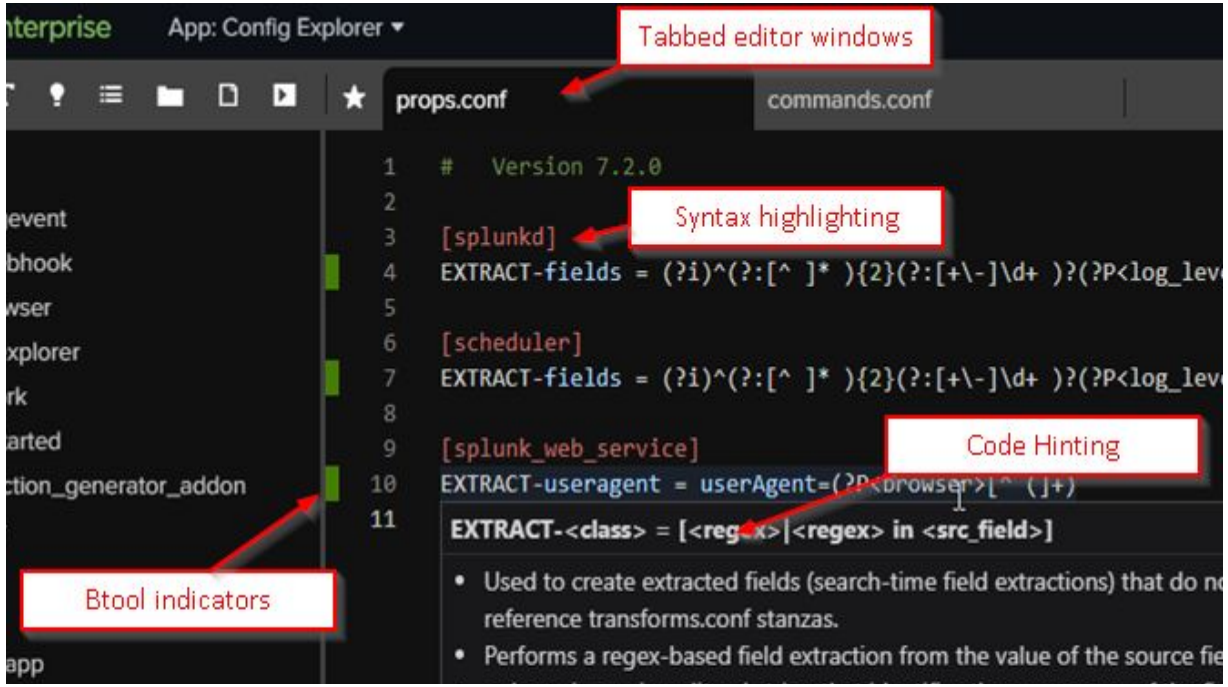
https://splunkbase.splunk.com/apps/#/product/all/author/chrisyoungerjds

What is Config Explorer

https://splunkbase.splunk.com/app/4353/

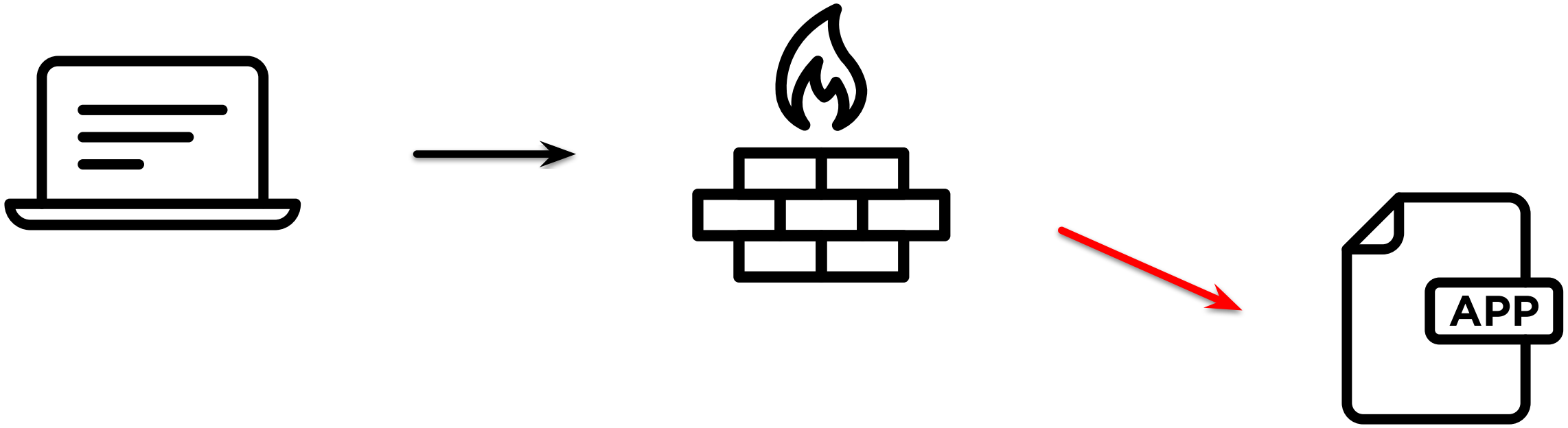
- Embeds the excellent Microsoft Monaco editor (VS Code)
- Code completion and tooltips
- Code gutter highlights using btool
- Displays the local spec files
- Diff files
- Works on Linux and Windows

- **Provides useful developer hooks (bump, debug refresh etc.) that can be customised and extended**
- **Optional version control of all changes by committing them to a git repository before and after changes**



The Magic

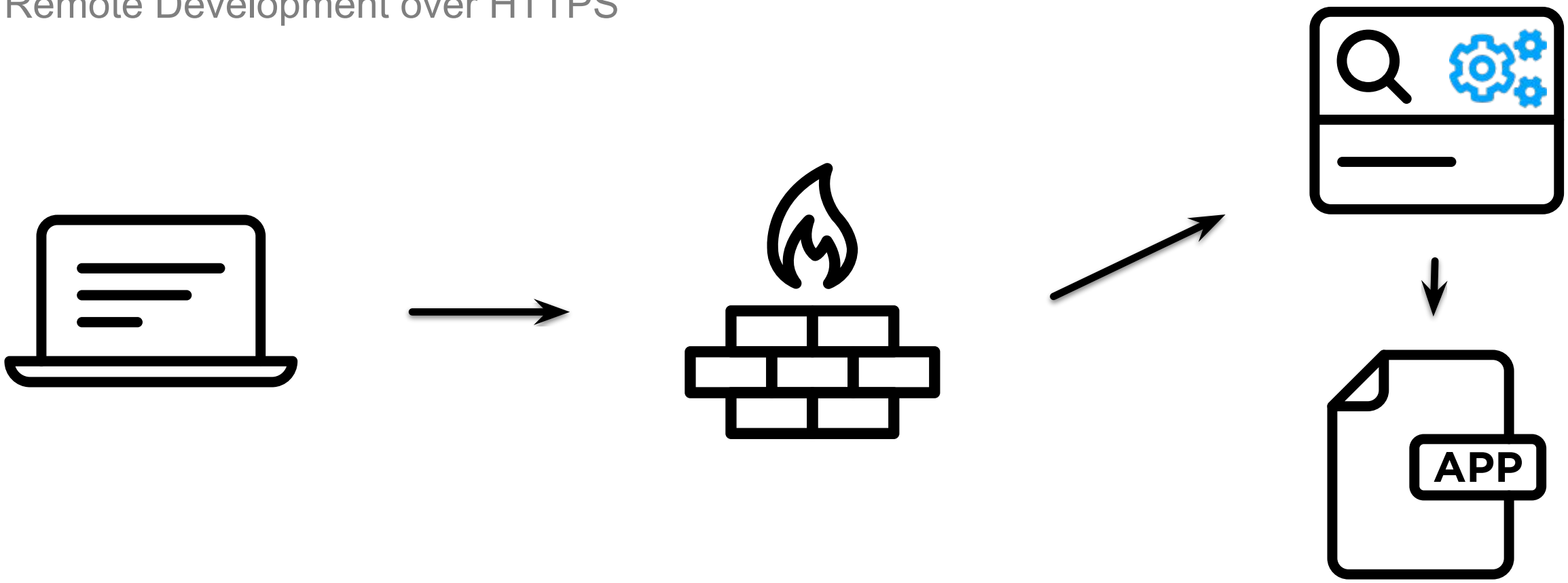
Remote Development over HTTPS



To learn more about developing with Visual Studio Code instead, check out **DEV1385** with Jason Conger.

The Magic

Remote Development over HTTPS



To learn more about developing with Visual Studio Code instead, check out **DEV1385** with Jason Conger.

Build in Enterprise

Run in Cloud & Enterprise

splunk> .conf22

Development Resources

https://dev.splunk.com/view/SP-CAAAER3

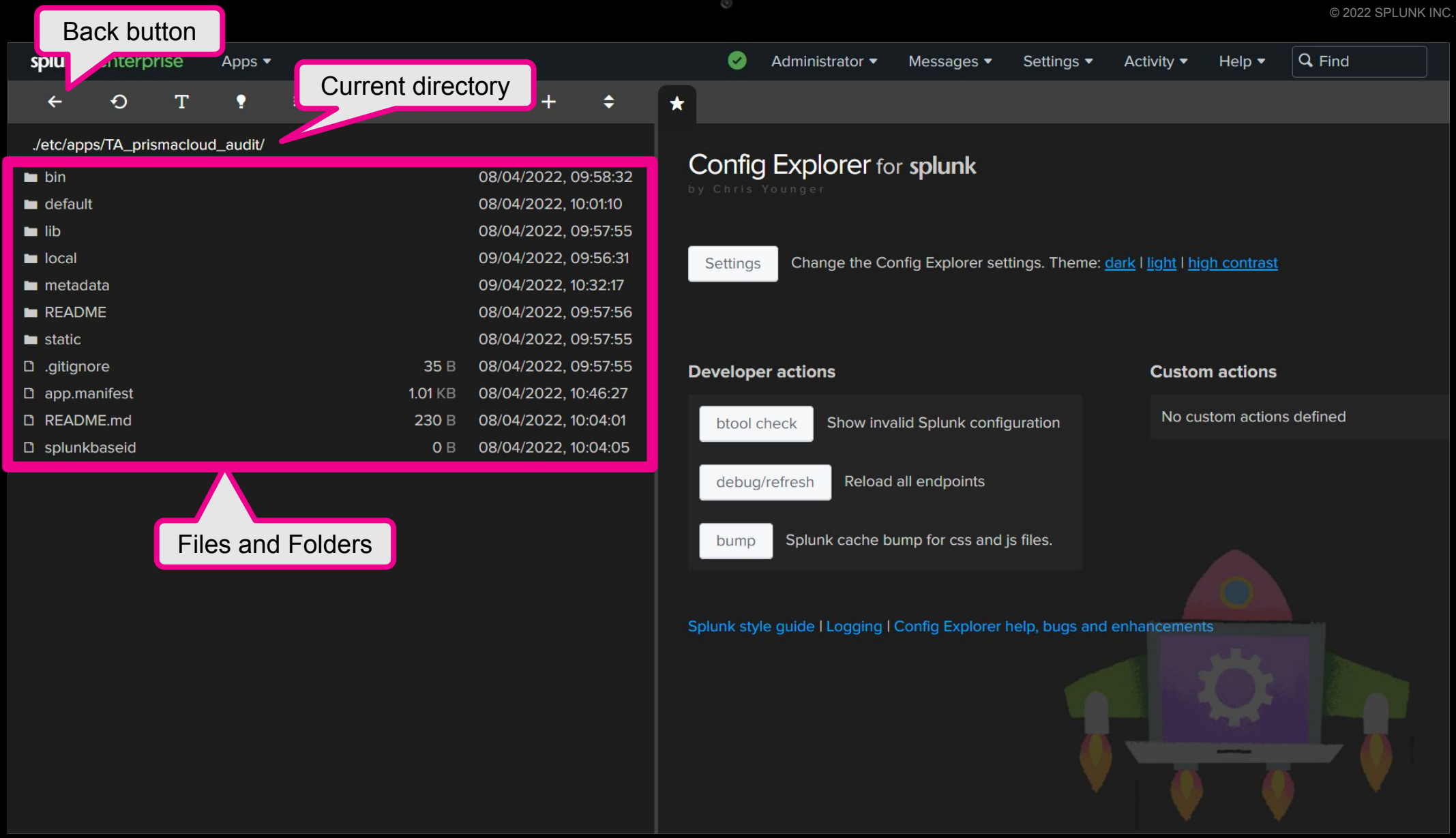
https://github.com/splunk/splunk-sdk-python/tree/master/examples/github_commits

https://github.com/LukeMurphey/splunk-modular-input

I recommend developing Splunk Apps with Splunk® Enterprise **running on Linux.**

splunk> .conf22

Let’s Build an App!



splunk> .conf22

splunk>enterprise Apps ▾ Administrator ▾ Messages ▾ Settings ▾ Activity ▾ Help ▾ Find

./etc/apps/TA_prismacloud_audit/

bin		08/04/2022, 09:58:32
default		08/04/2022, 10:01:10
lib		08/04/2022, 09:57:55
local		09/04/2022, 09:56:31
metadata		09/04/2022, 10:32:17
README		08/04/2022, 09:57:56
static		08/04/2022, 09:57:55
.gitignore	35 B	08/04/2022, 09:57:55
app.manifest	1.01 KB	08/04/2022, 10:46:27
README.md	230 B	08/04/2022, 10:04:01
splunkbaseid	0 B	08/04/2022, 10:04:05

Config Explorer for splunk

by Chris Younger

Settings

Change the Config Explorer settings. Theme: dark | light | high contrast

Opens the Config Explorer Settings

Developer actions

btool check — Show invalid Splunk configuration

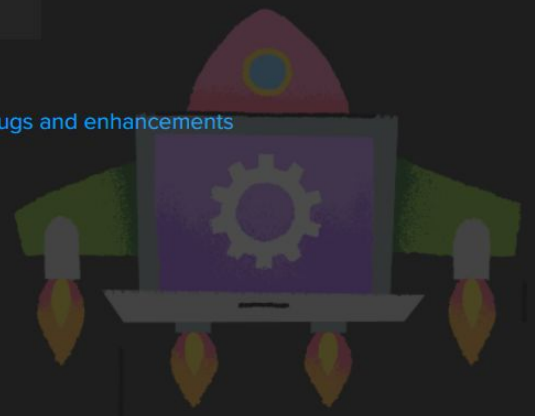
debug/refresh — Reload all endpoints

bump — Splunk cache bump for css and js files.

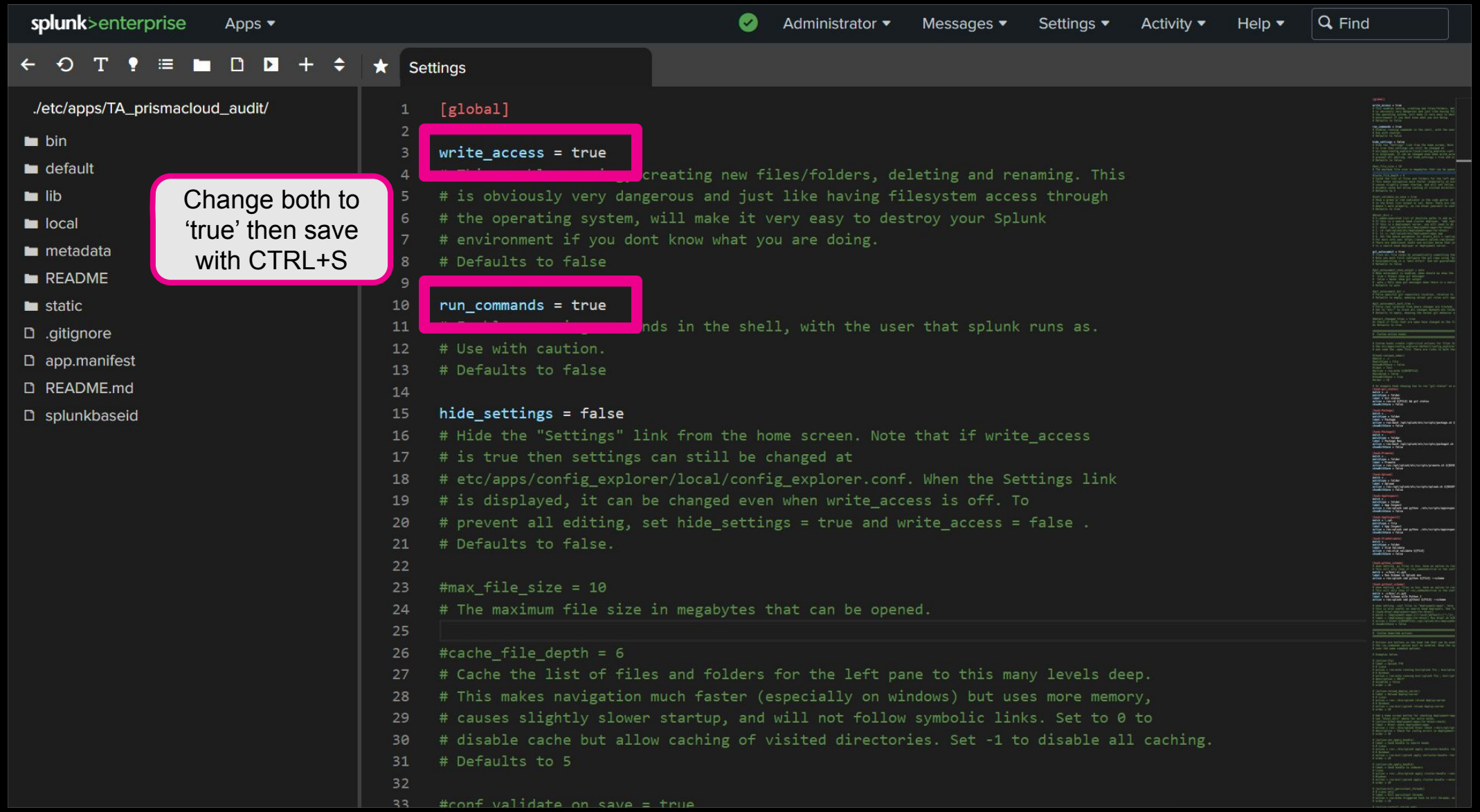
Custom actions

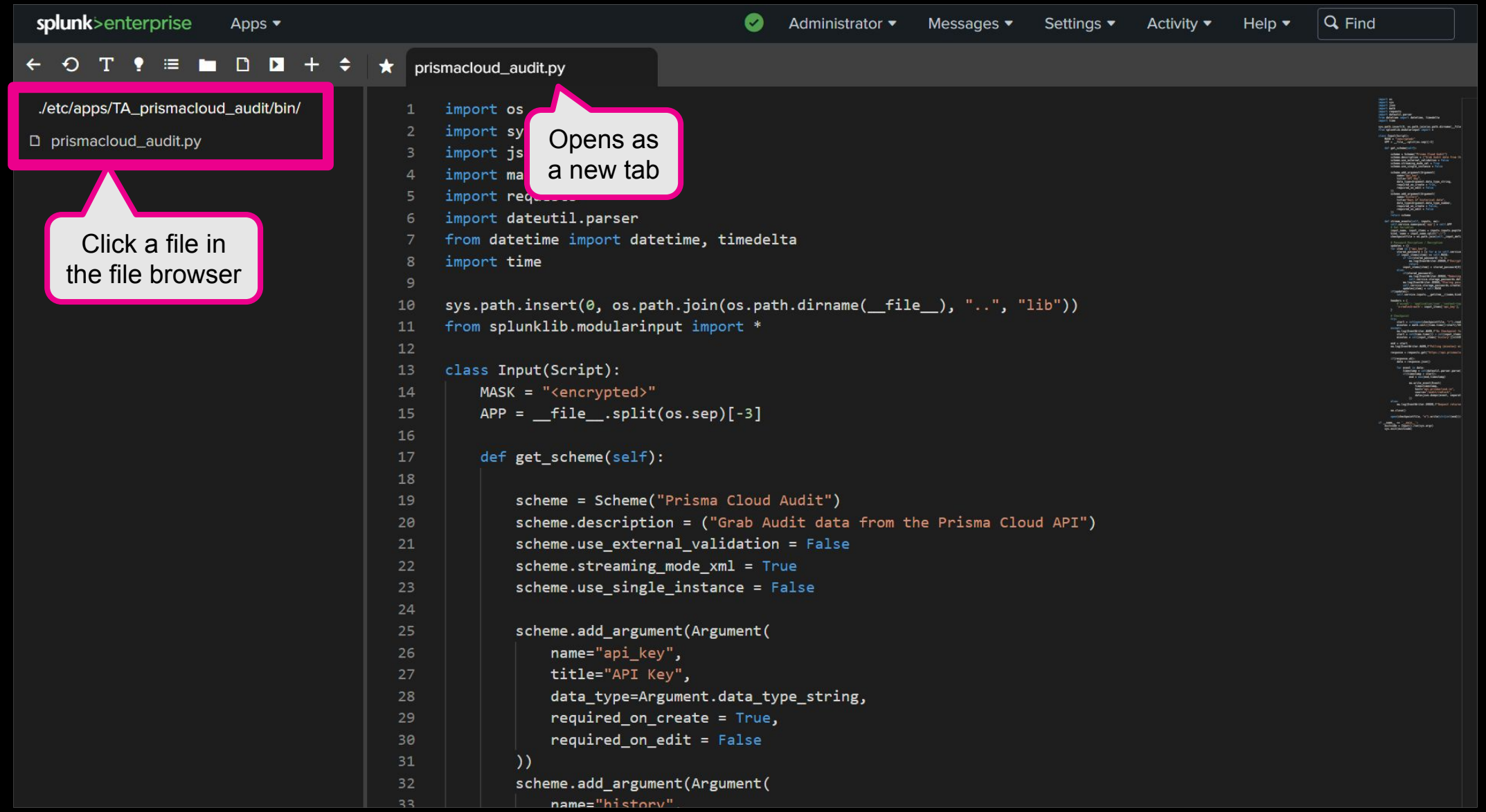
No custom actions defined

Splunk style guide | Logging | Config Explorer help, bugs and enhancements



splunk> .conf22

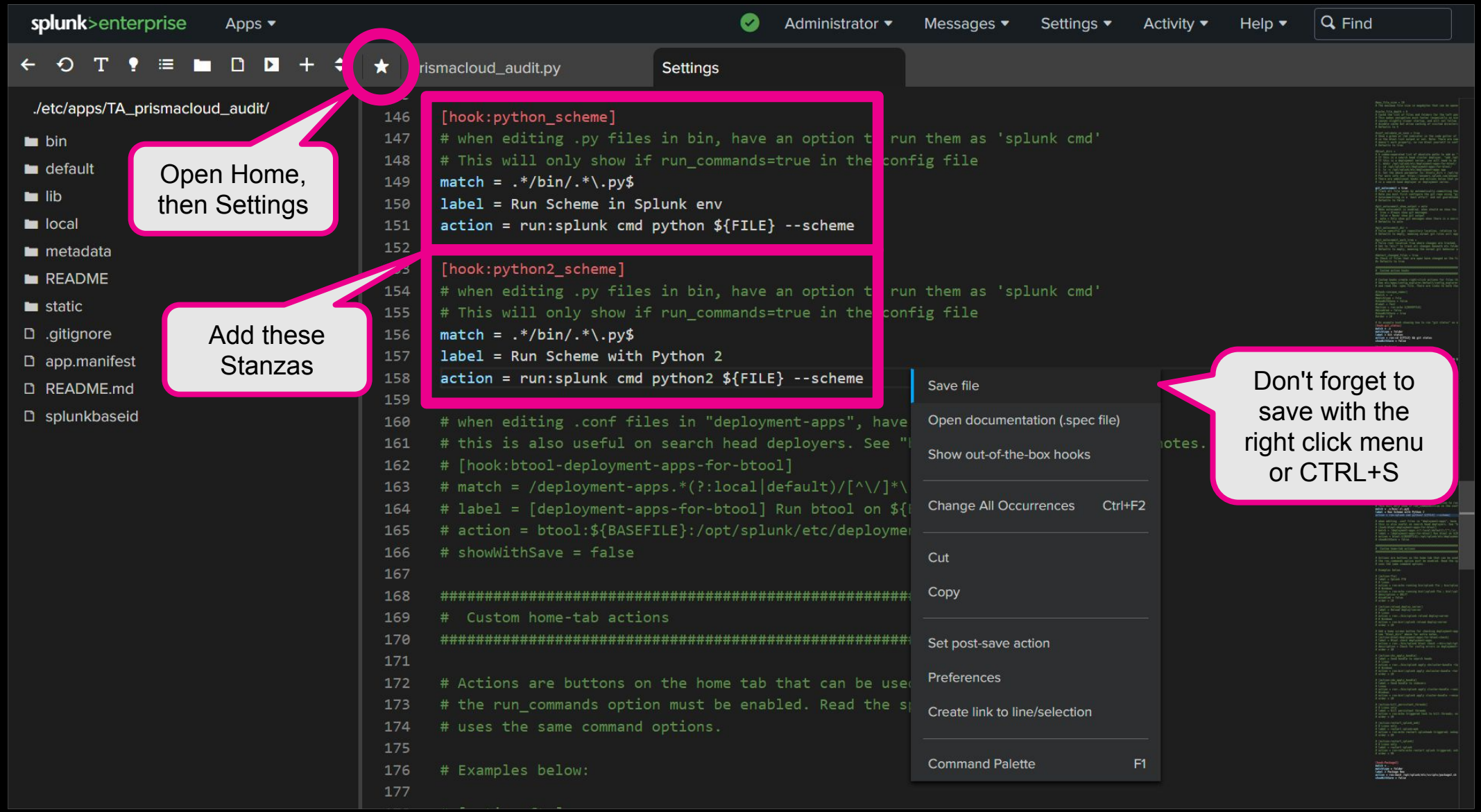




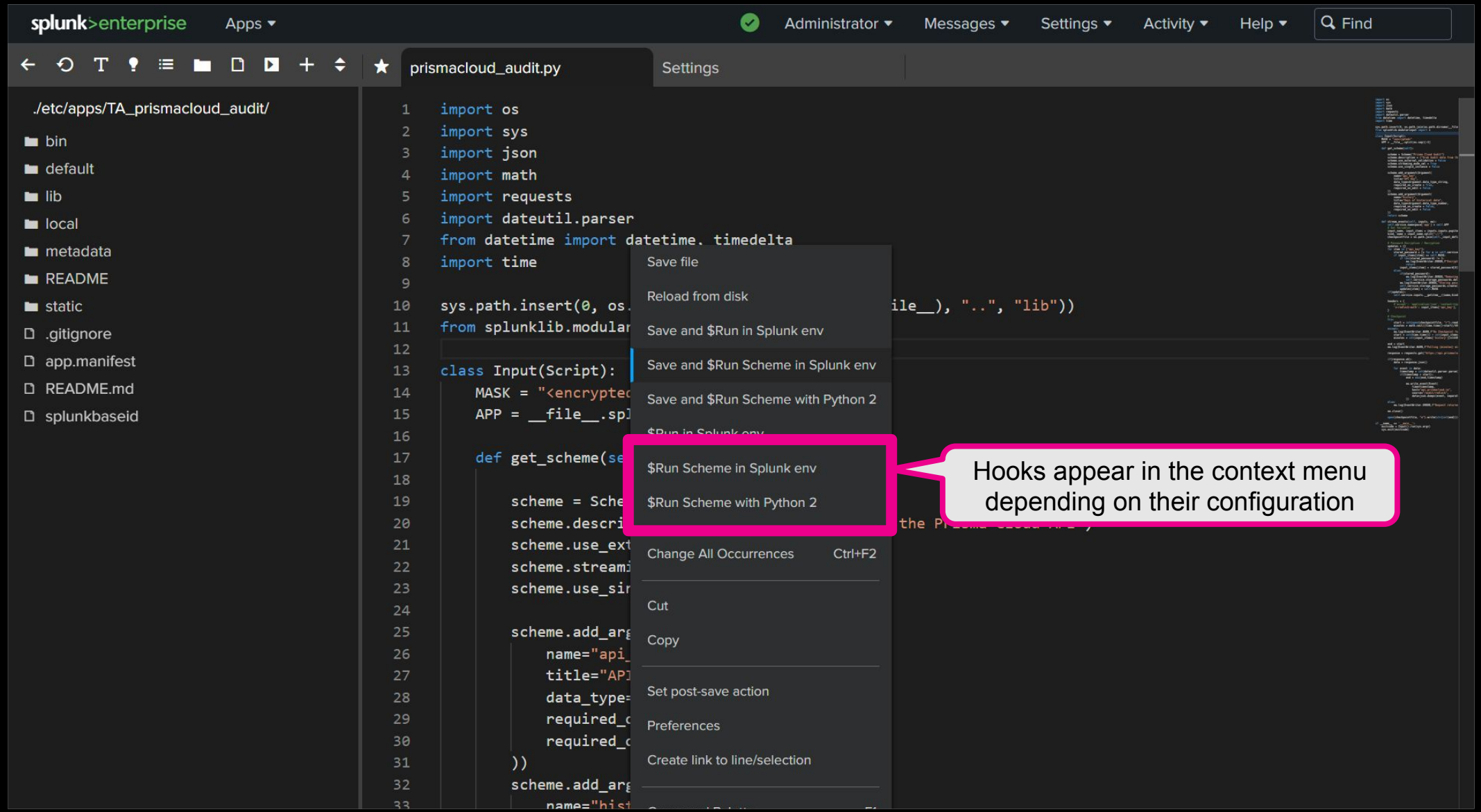
splunk> .conf22

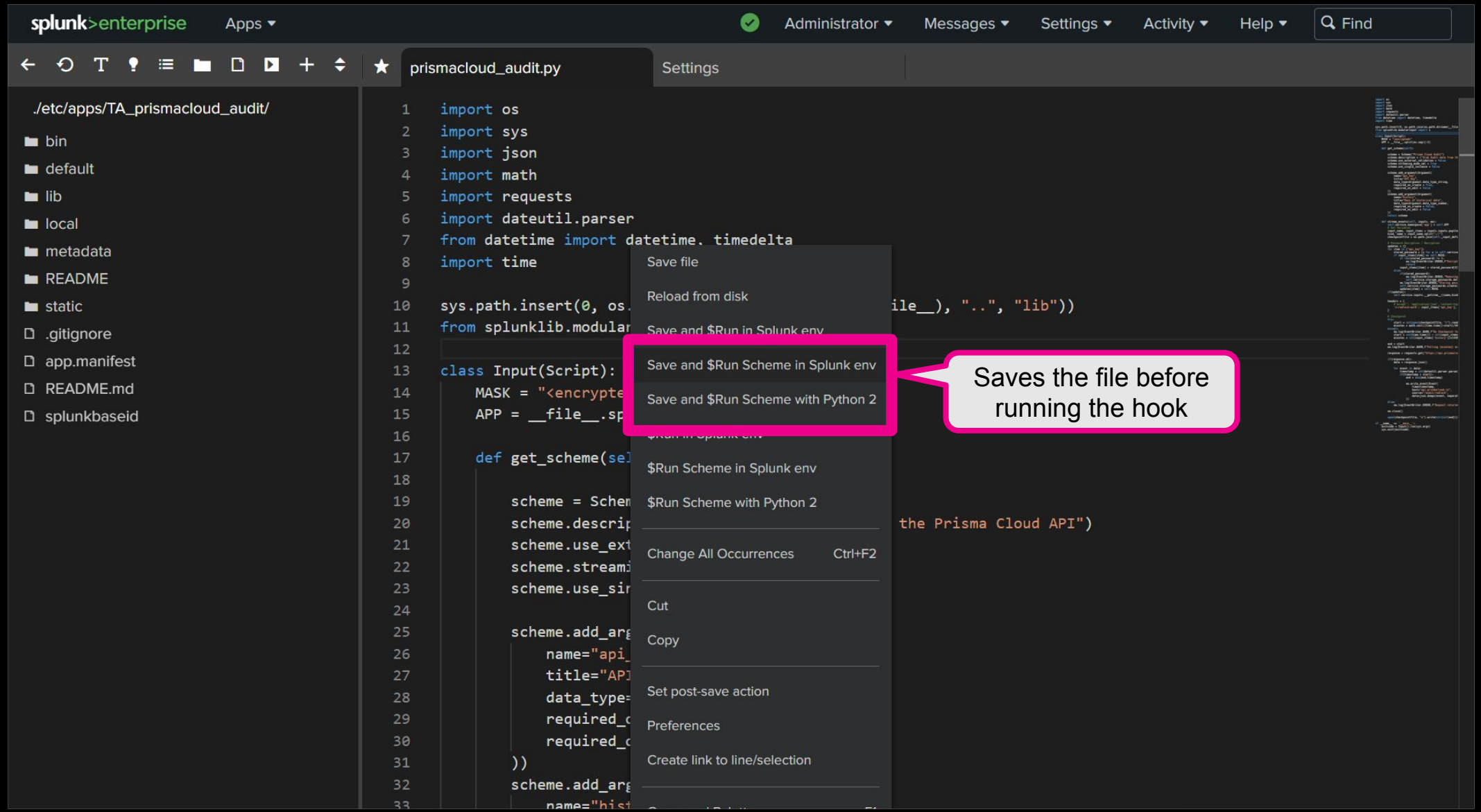
Hooks

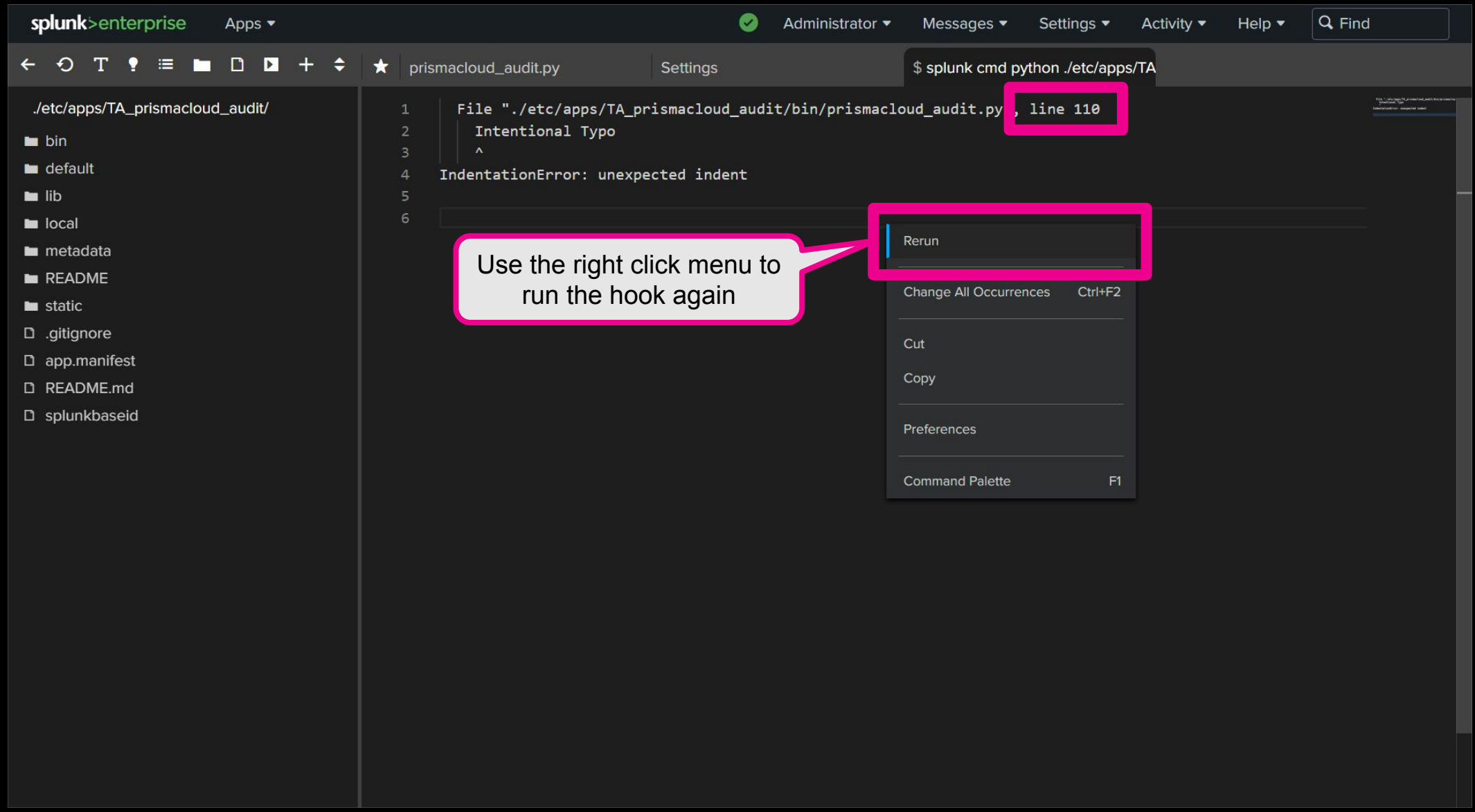
Being lazy for maximum productivity

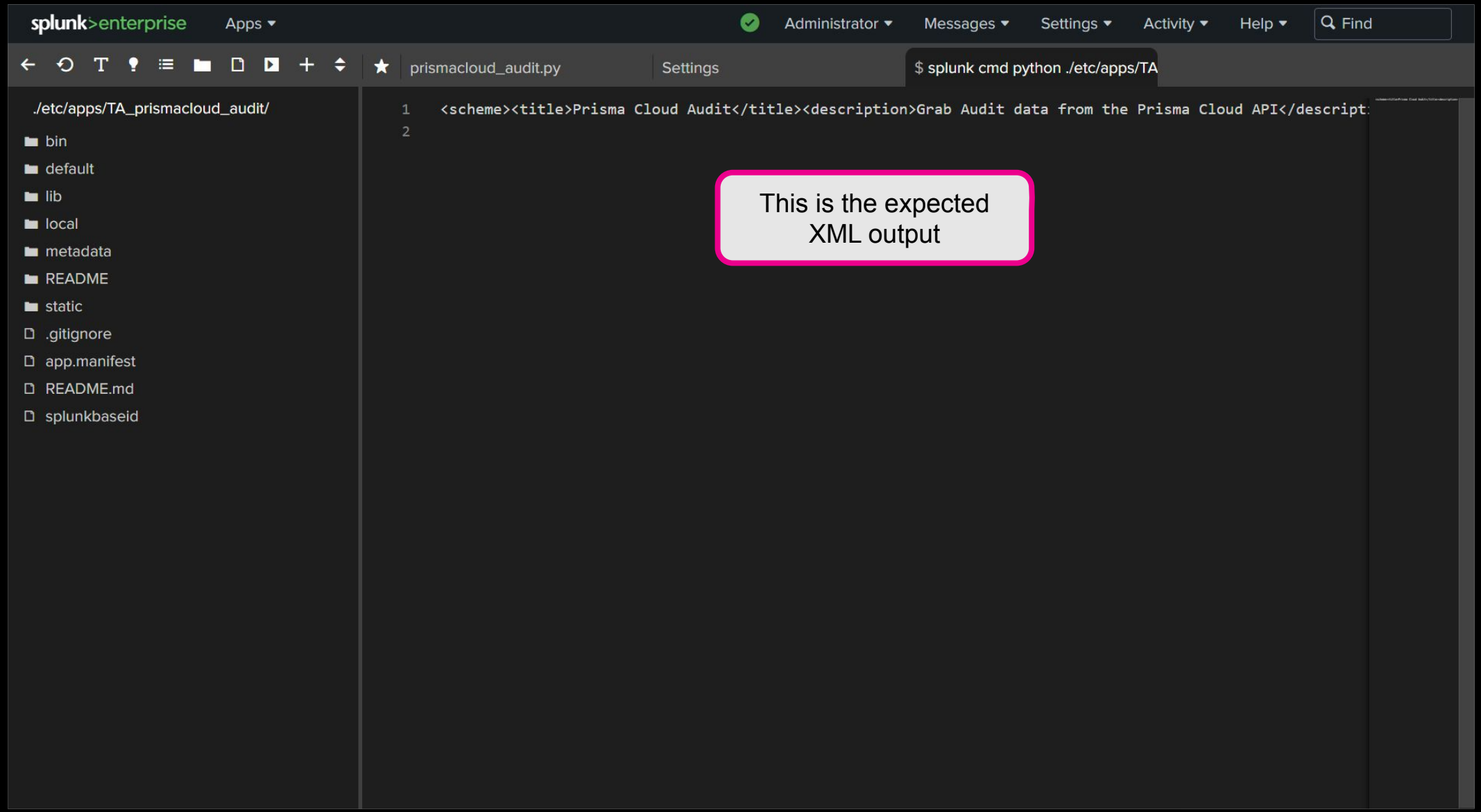


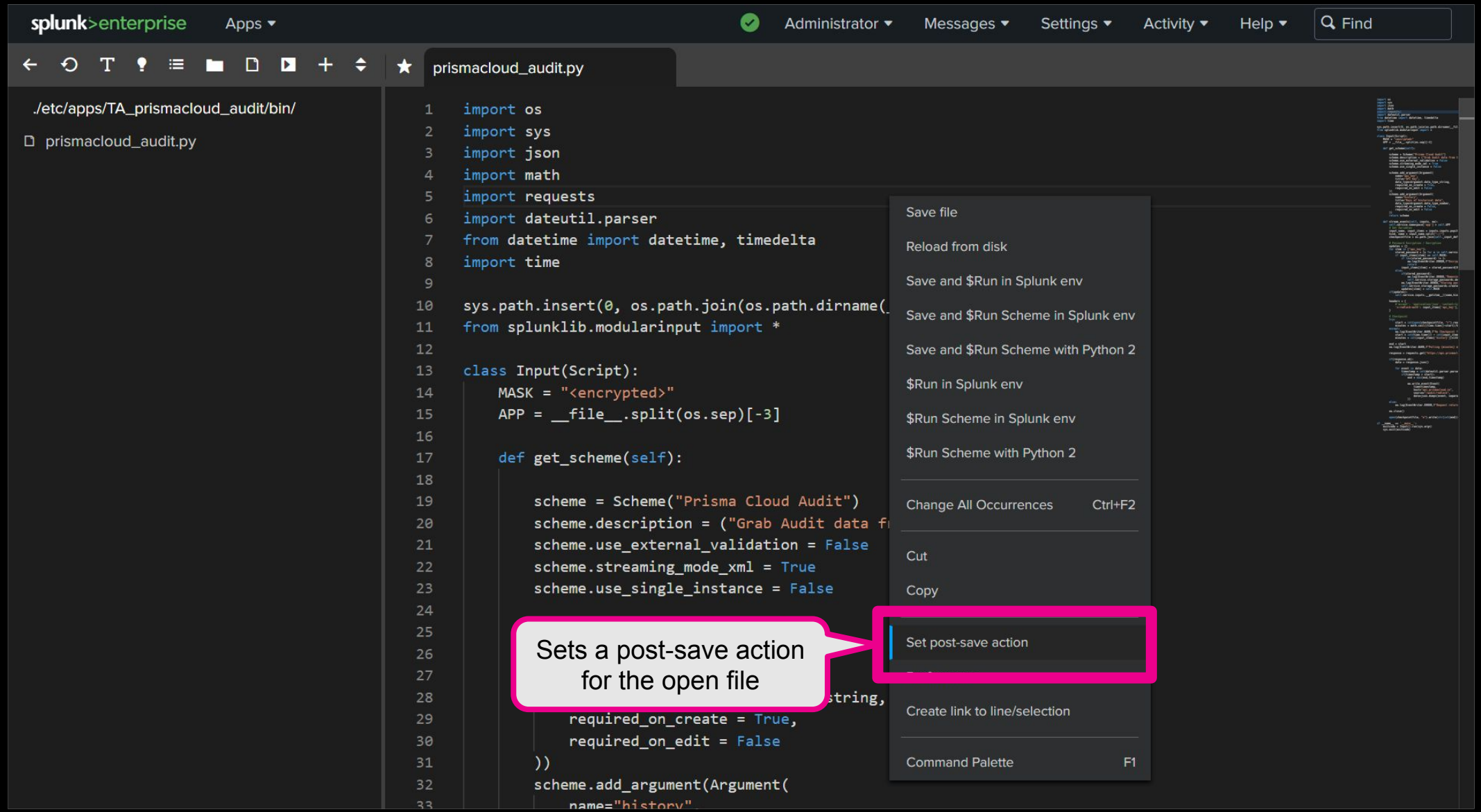
splunk> .conf22

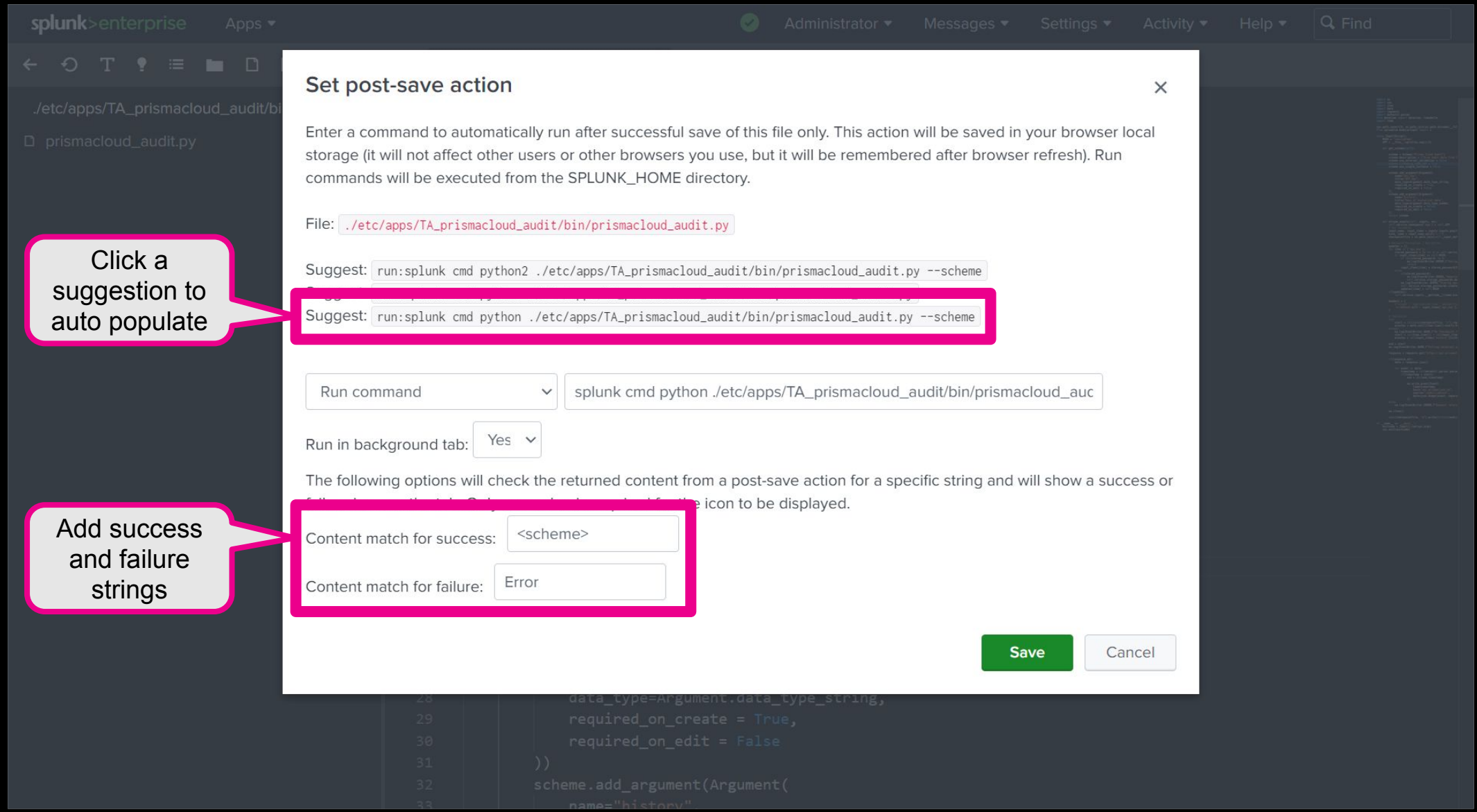




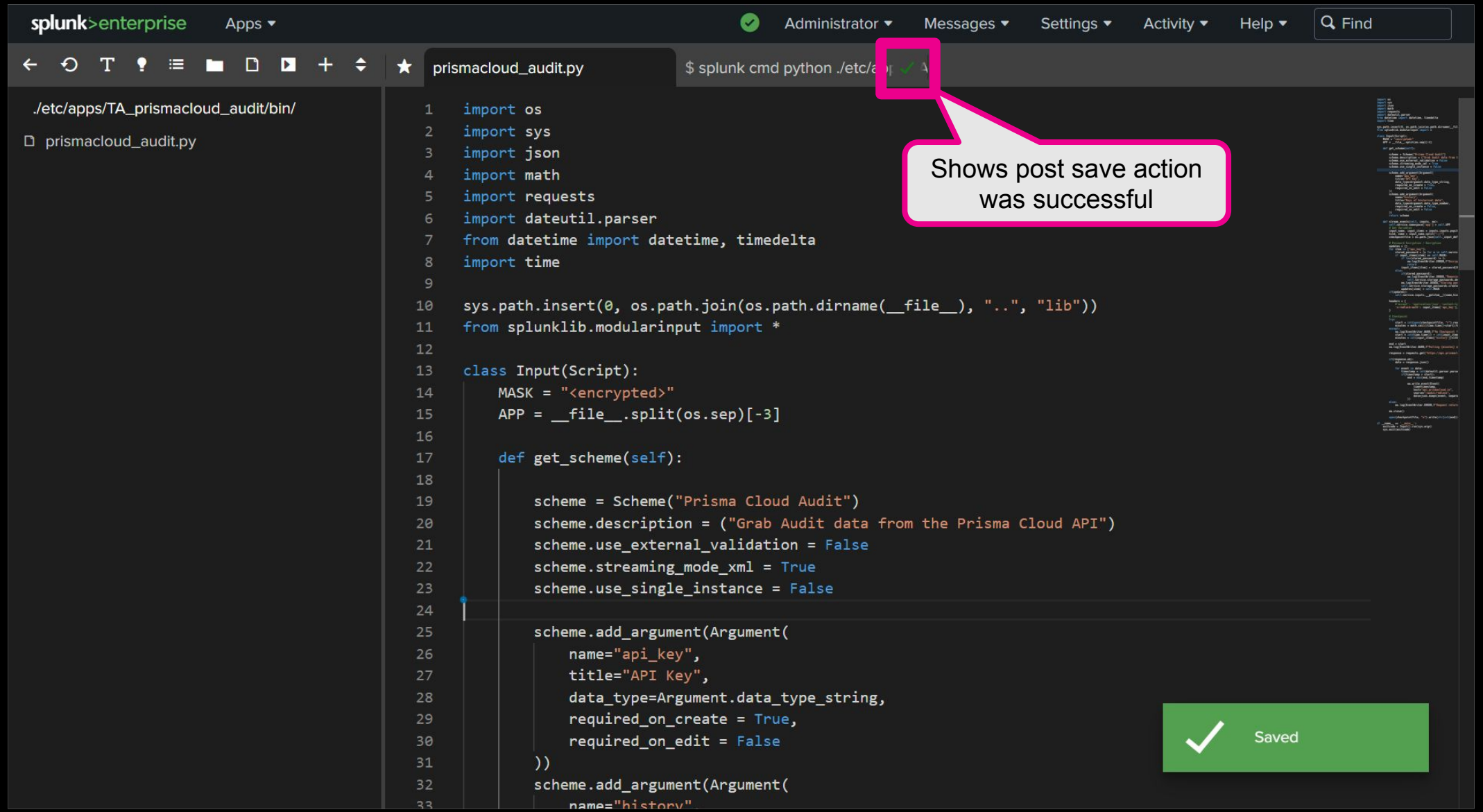








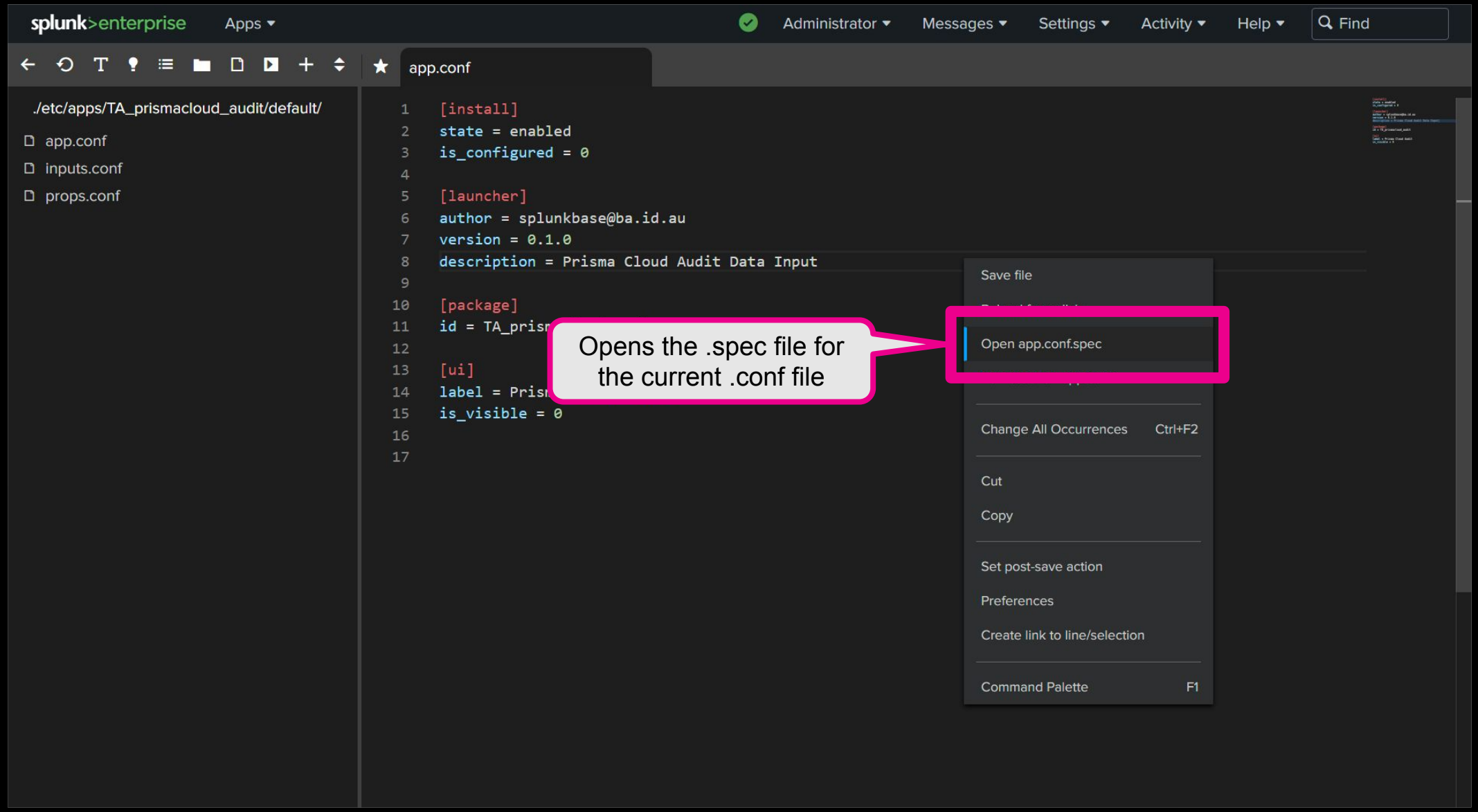
splunk> .conf22

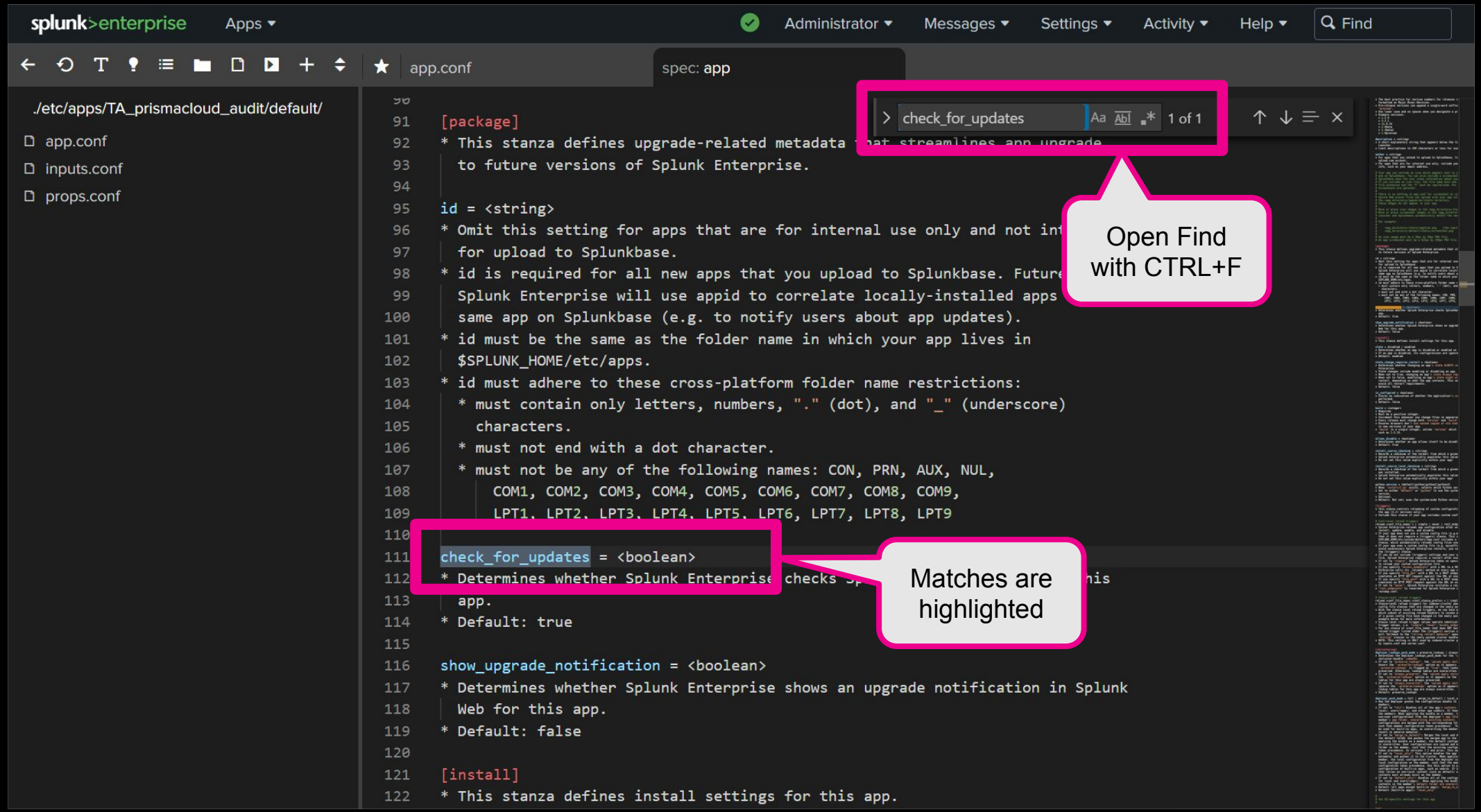


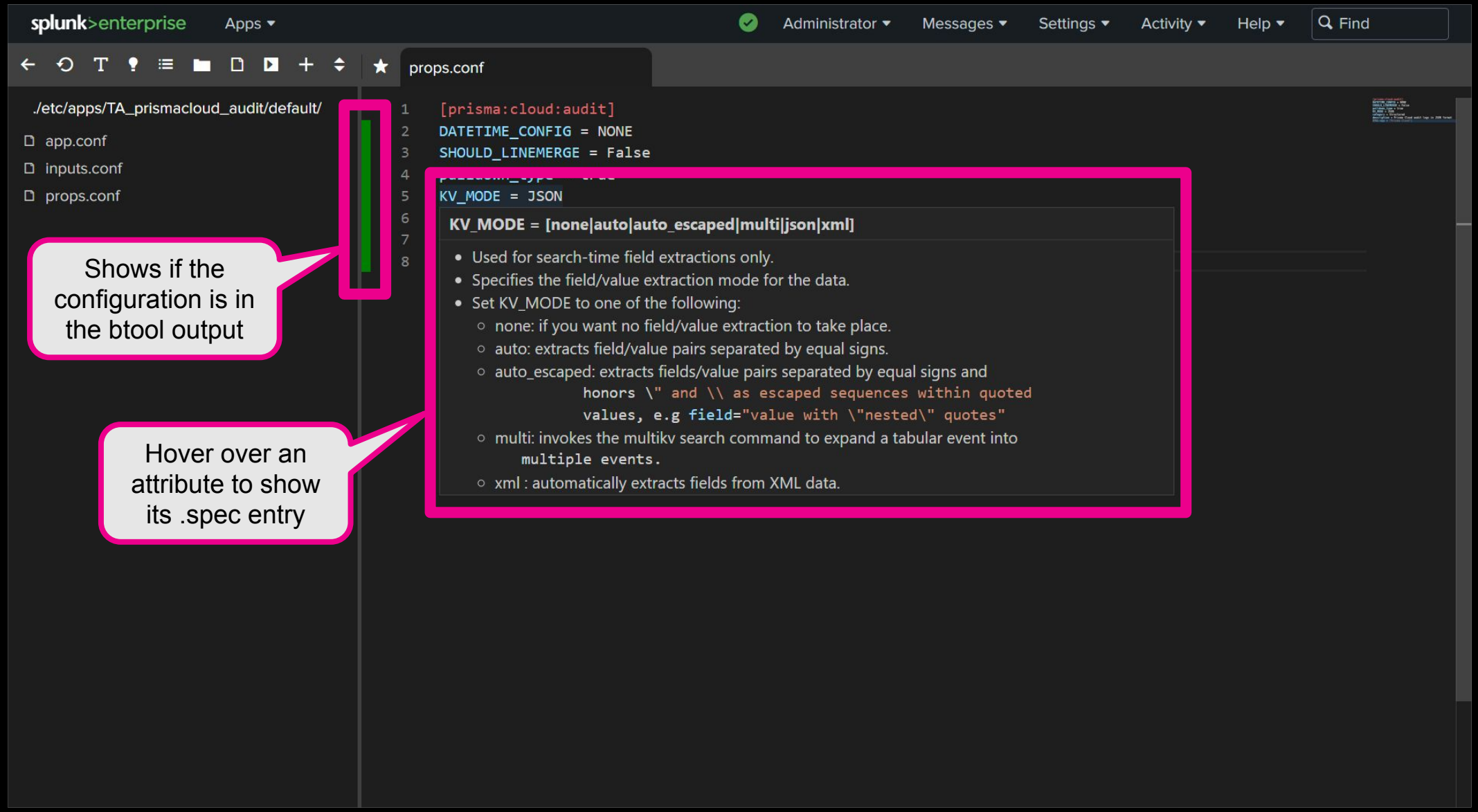

```python
import os
import sys
import json
import math
import requests
import dateutil.parser
from datetime import datetime, timedelta
import time

sys.path.insert(0, os.path.join(os.path.dirname(__file__), "..", "lib"))
from splunklib.modularinput import *

class Input(Script):
    MASK = "<encrypted>"
    APP = __file__.split(os.sep)[-3]

    def get_scheme(self):

        scheme = Scheme("Prisma Cloud Audit")
        scheme.description = ("Grab Audit data from the Prisma Cloud API")
        scheme.use_external_validation = False
        scheme.streaming_mode_xml = True
        scheme.use_single_instance = False

        scheme.add_argument(Argument(
            name="api_key",
            title="API Key",
            data_type=Argument.data_type_string,
            required_on_create = True,
            required_on_edit = False
        ))
        scheme.add_argument(Argument(
            name="history"
```

Spec Files

Better than typing inputs.conf.spec into Google







splunk> .conf22

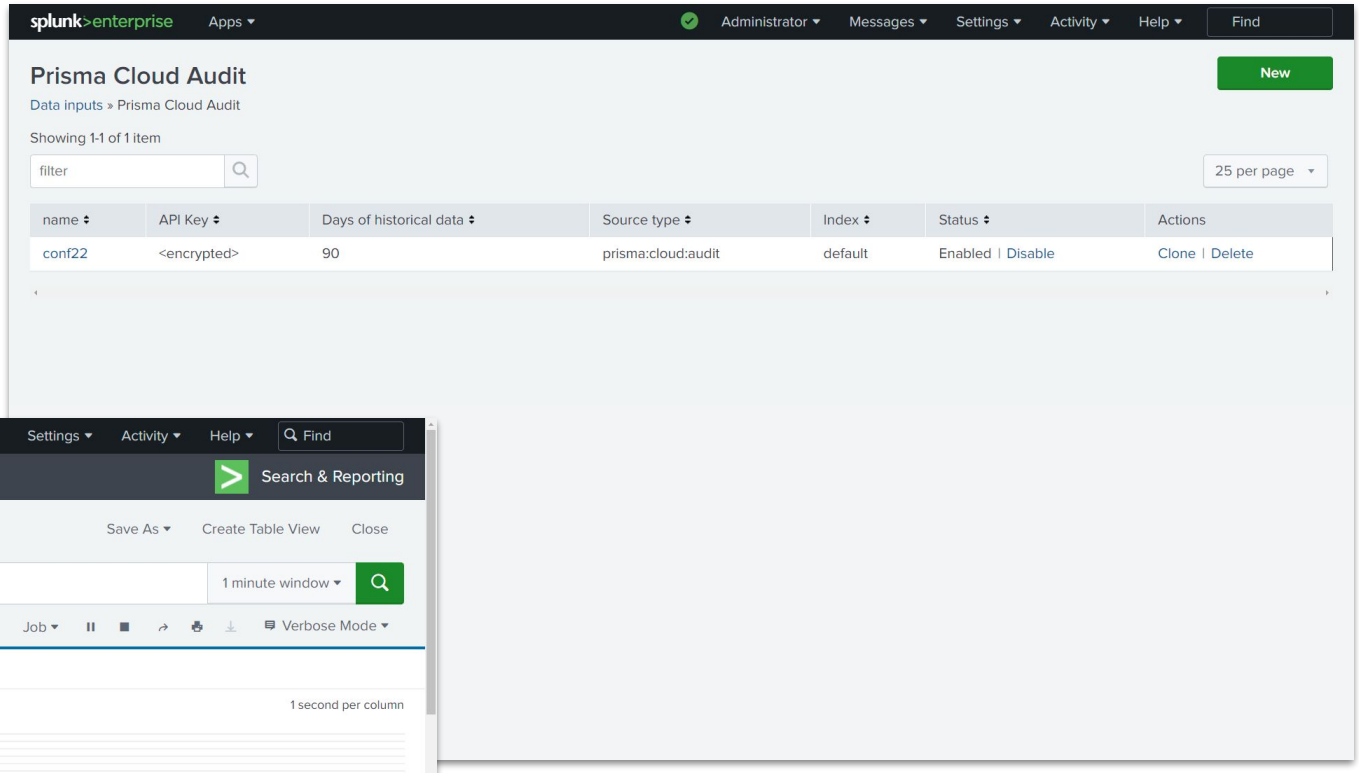
Testing

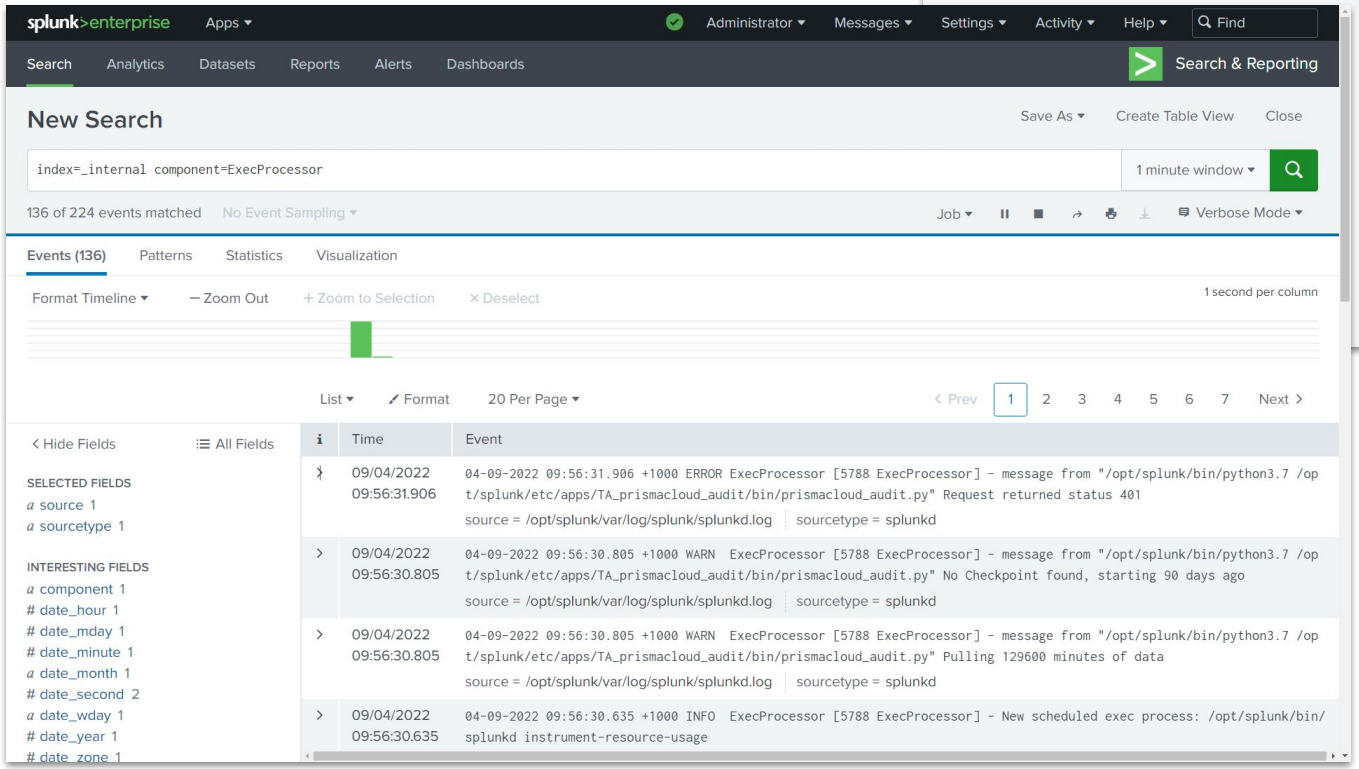
Finding your faults, just like mom

splunk> .conf22

Testing

Outside Config Explorer





splunk> .conf22

Perfect Packaging

The shell script way

Remove compiled and cached python files

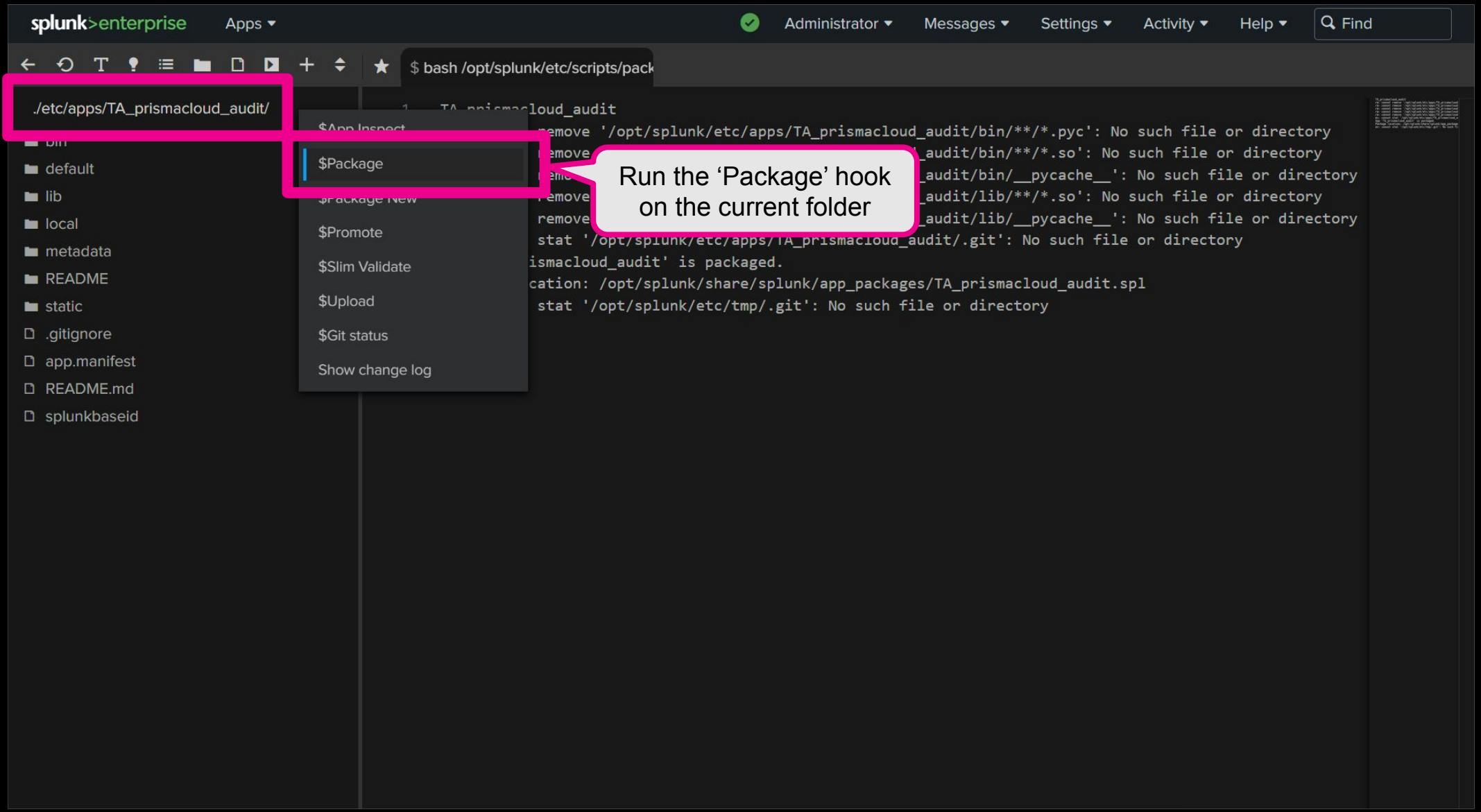
Remove metadata/local.meta

Remove any boilerplate files and Lookup Editor backups

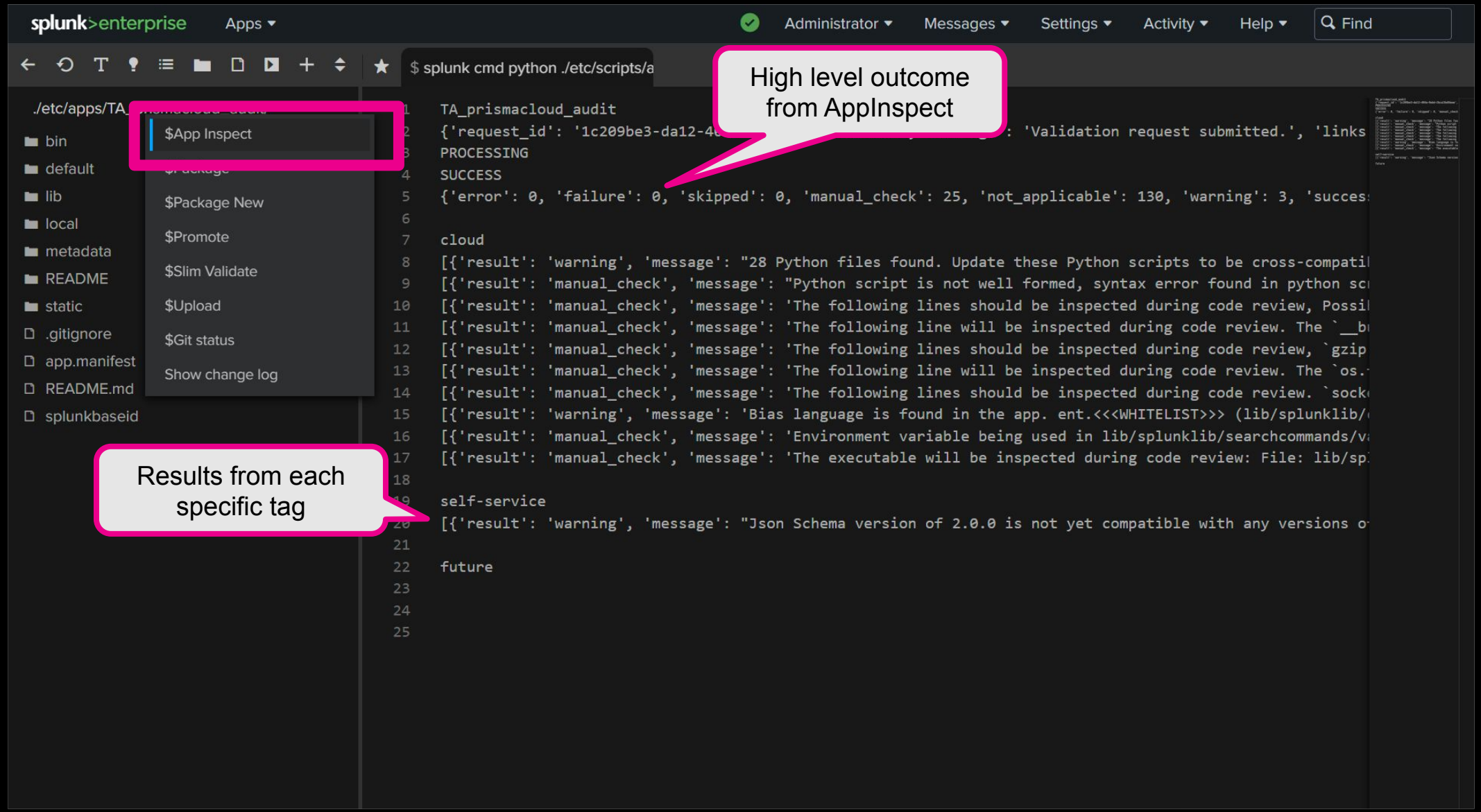
Set all files to 644 (Owner Read & Write, Others Read)

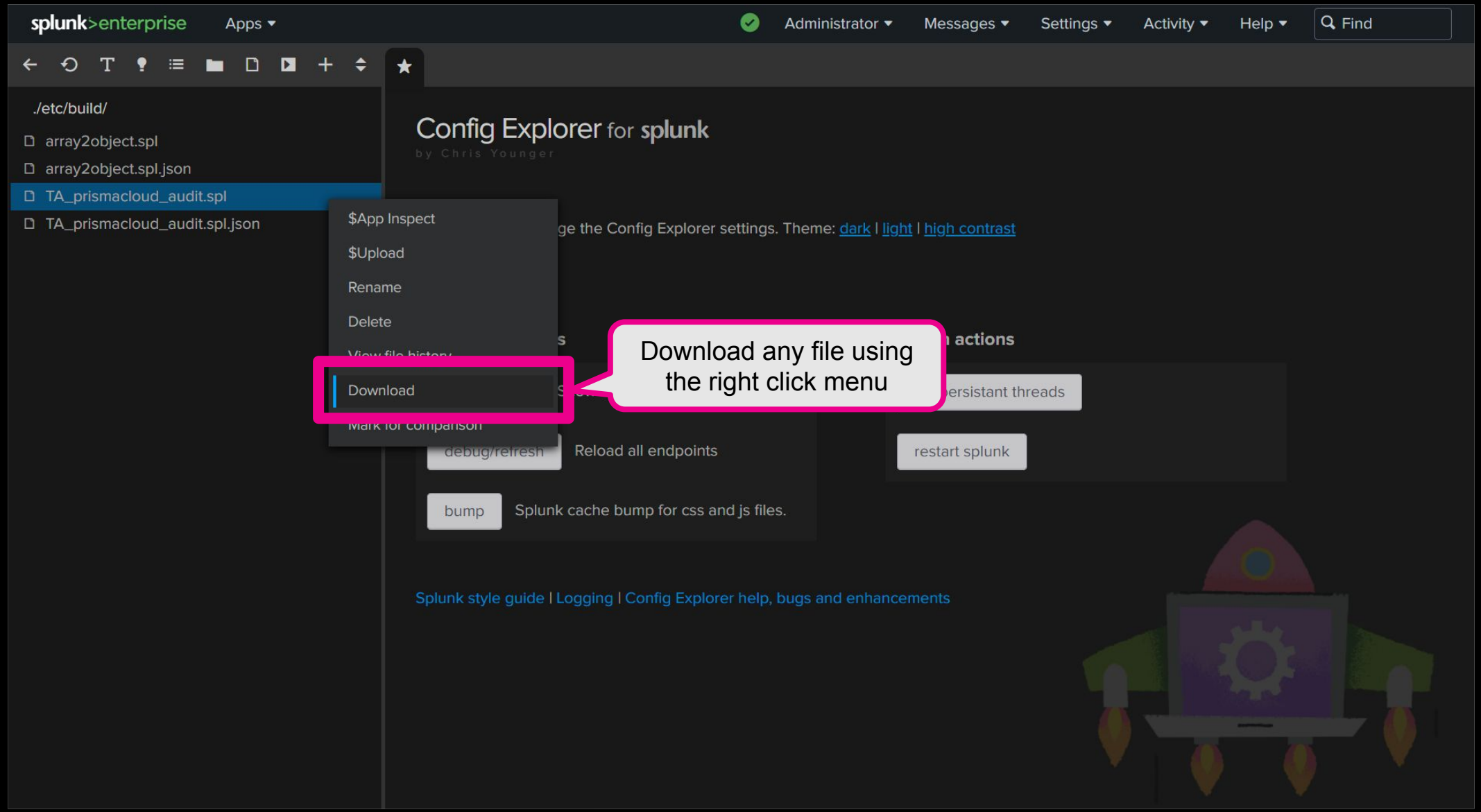
Add execute permission to folders and files in bin

Compress to tar gzip archive with .spl extension

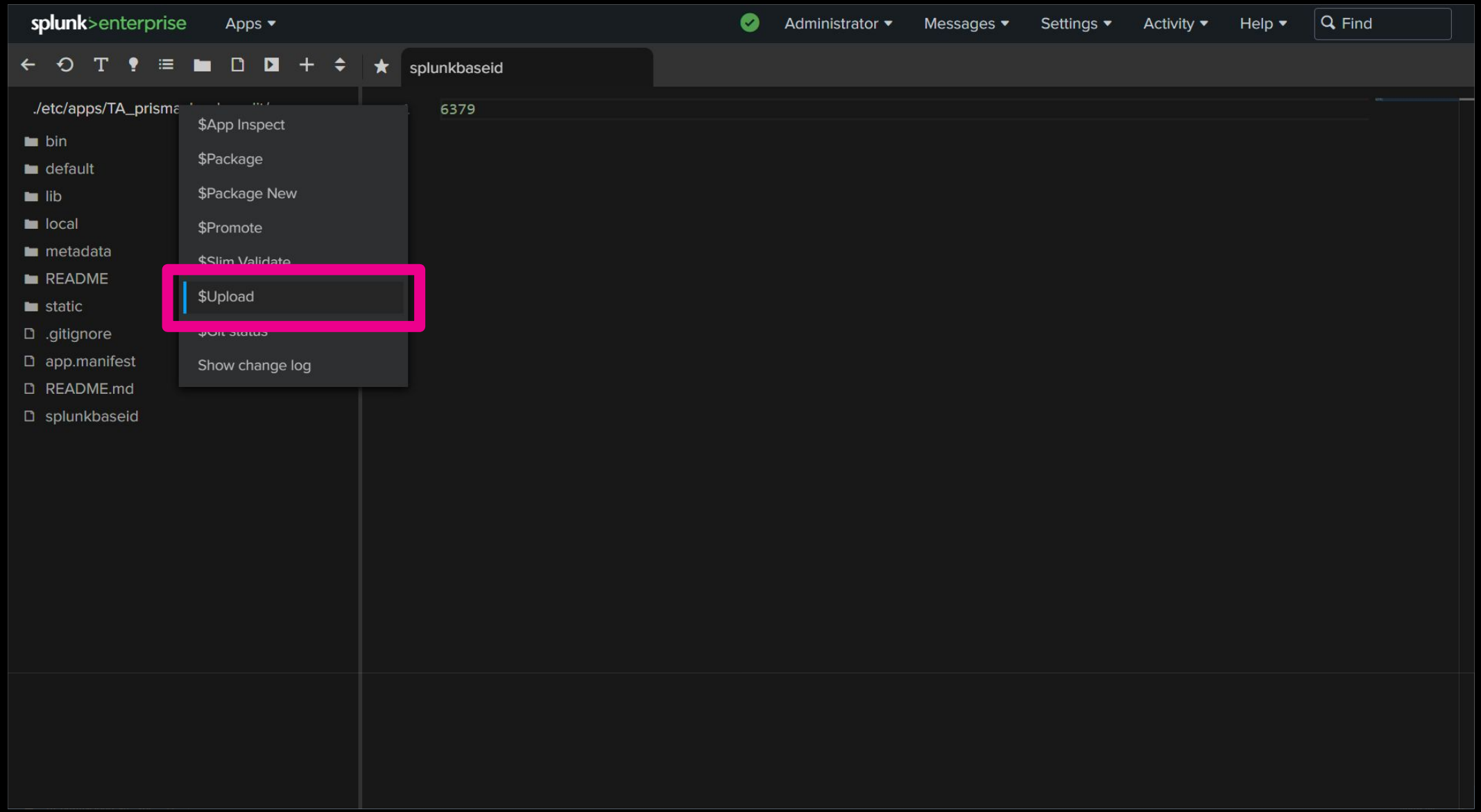


splunk> .conf22



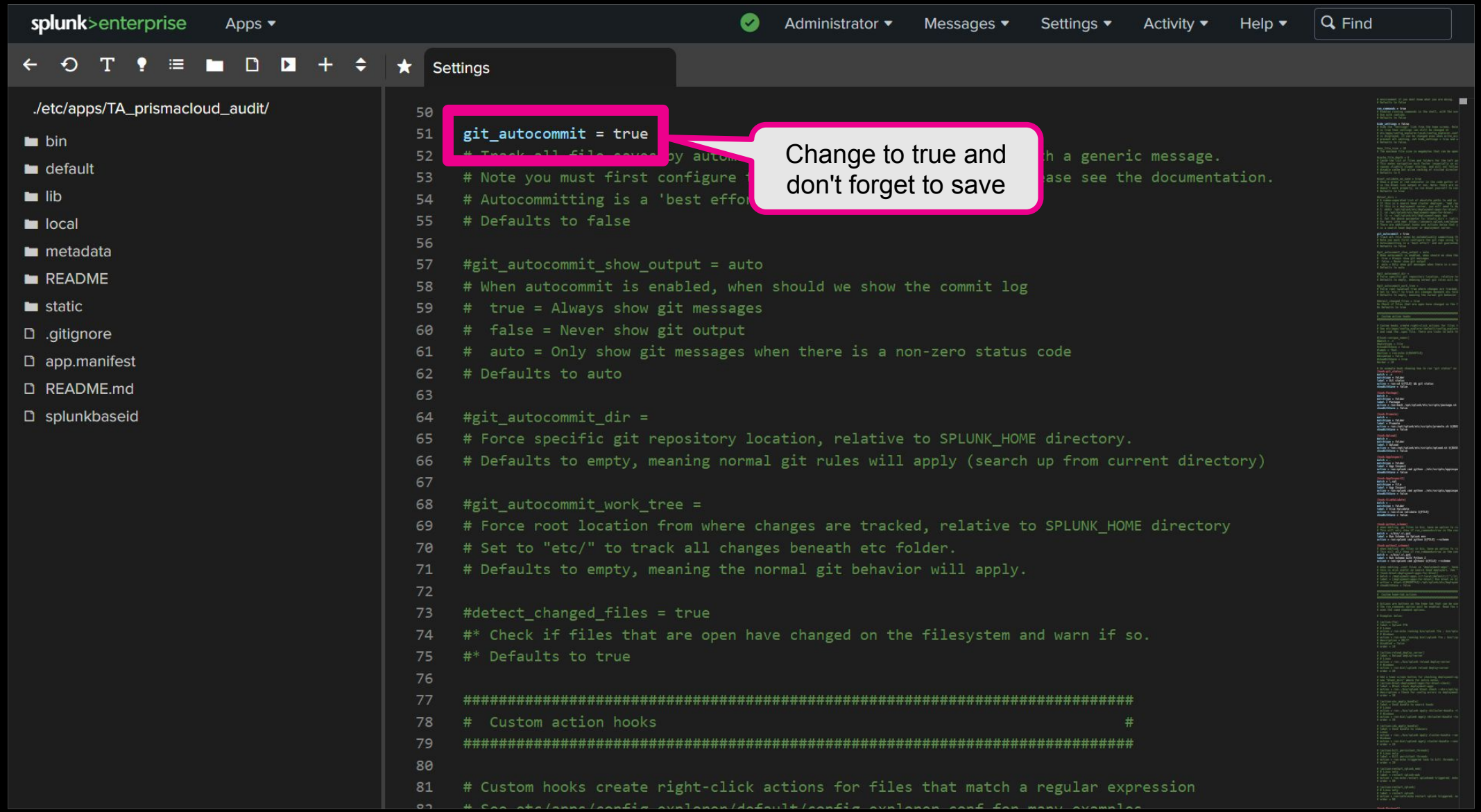


splunk> .conf22



Source Control

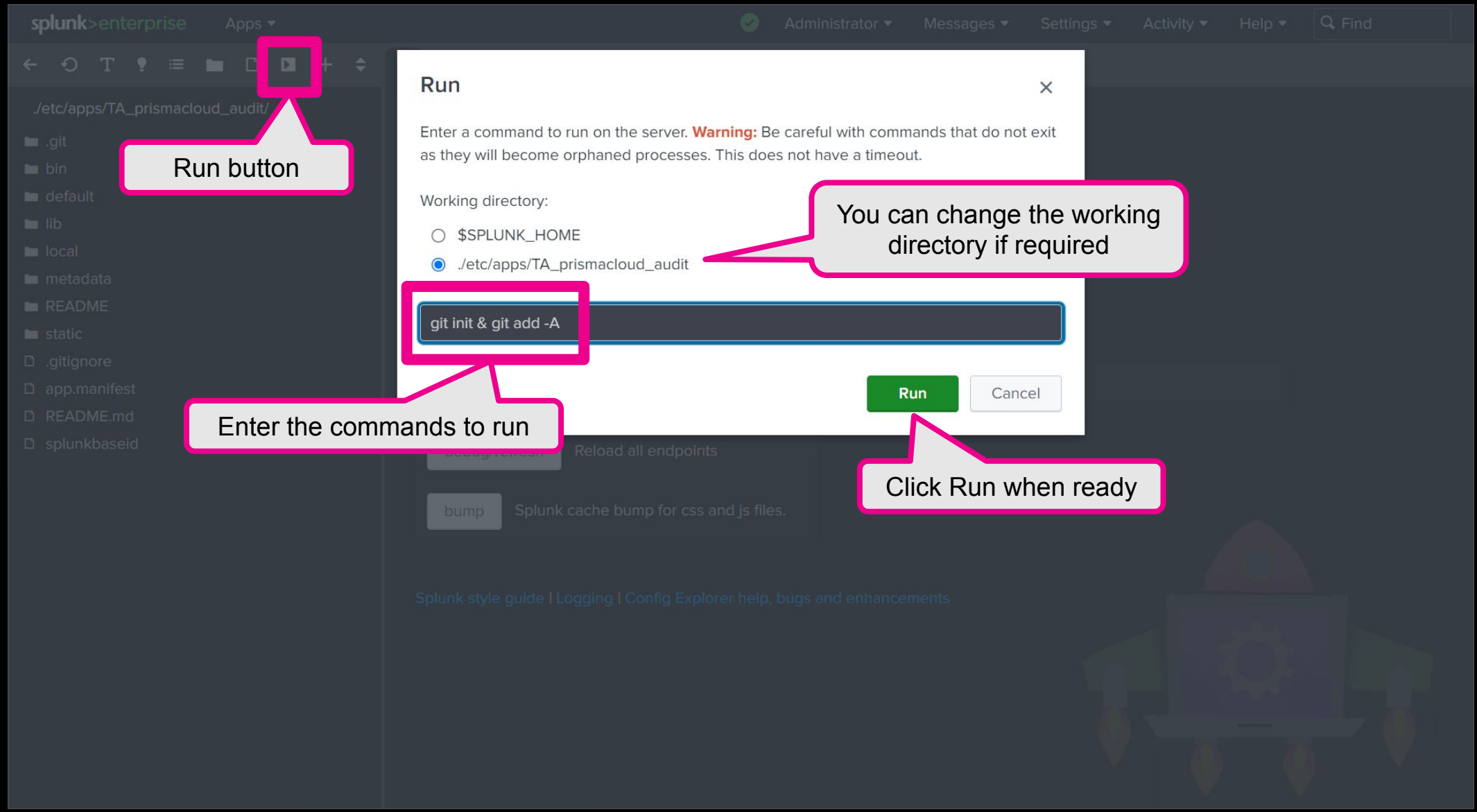
Keeping a permanent record of every mistake

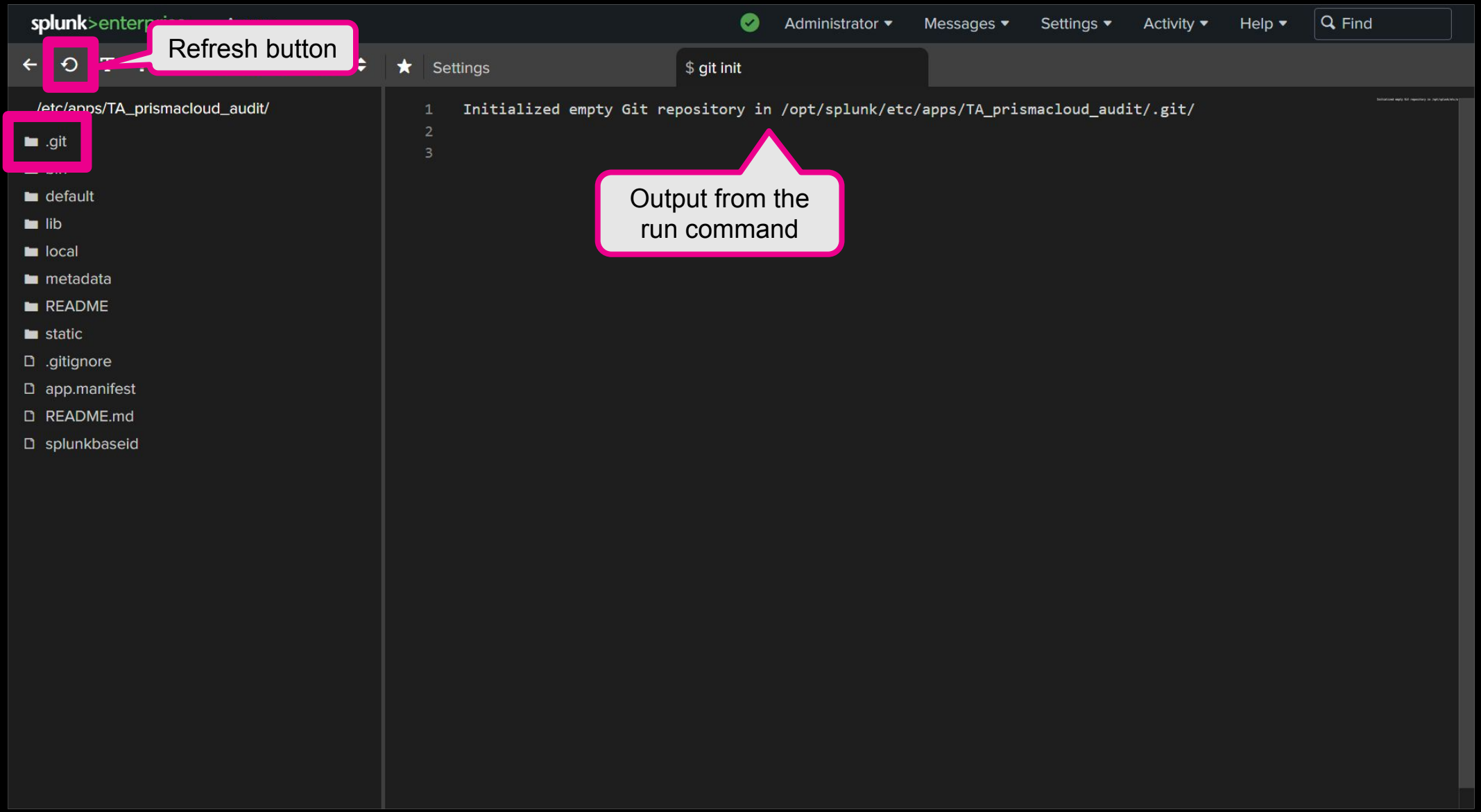


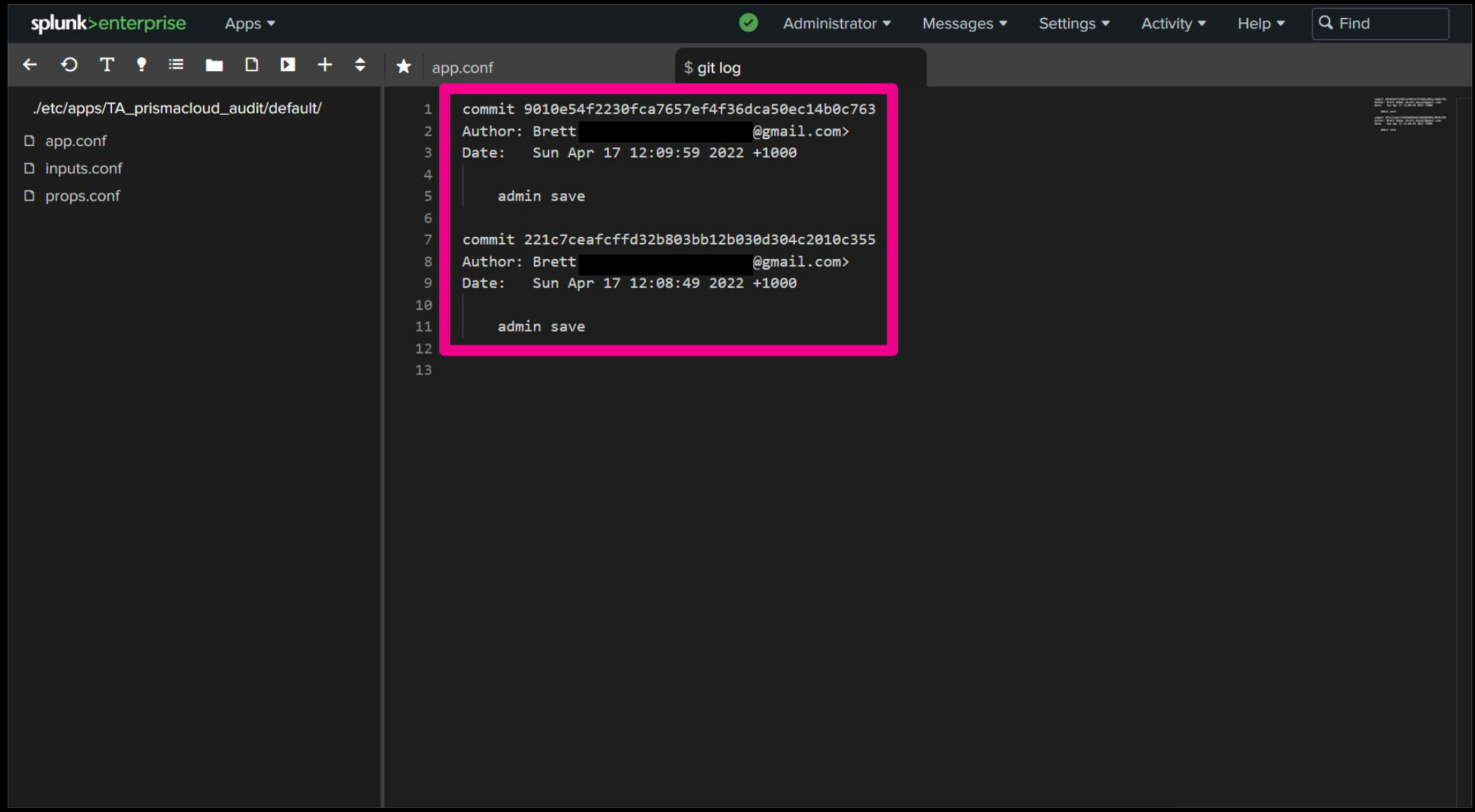
.gitignore

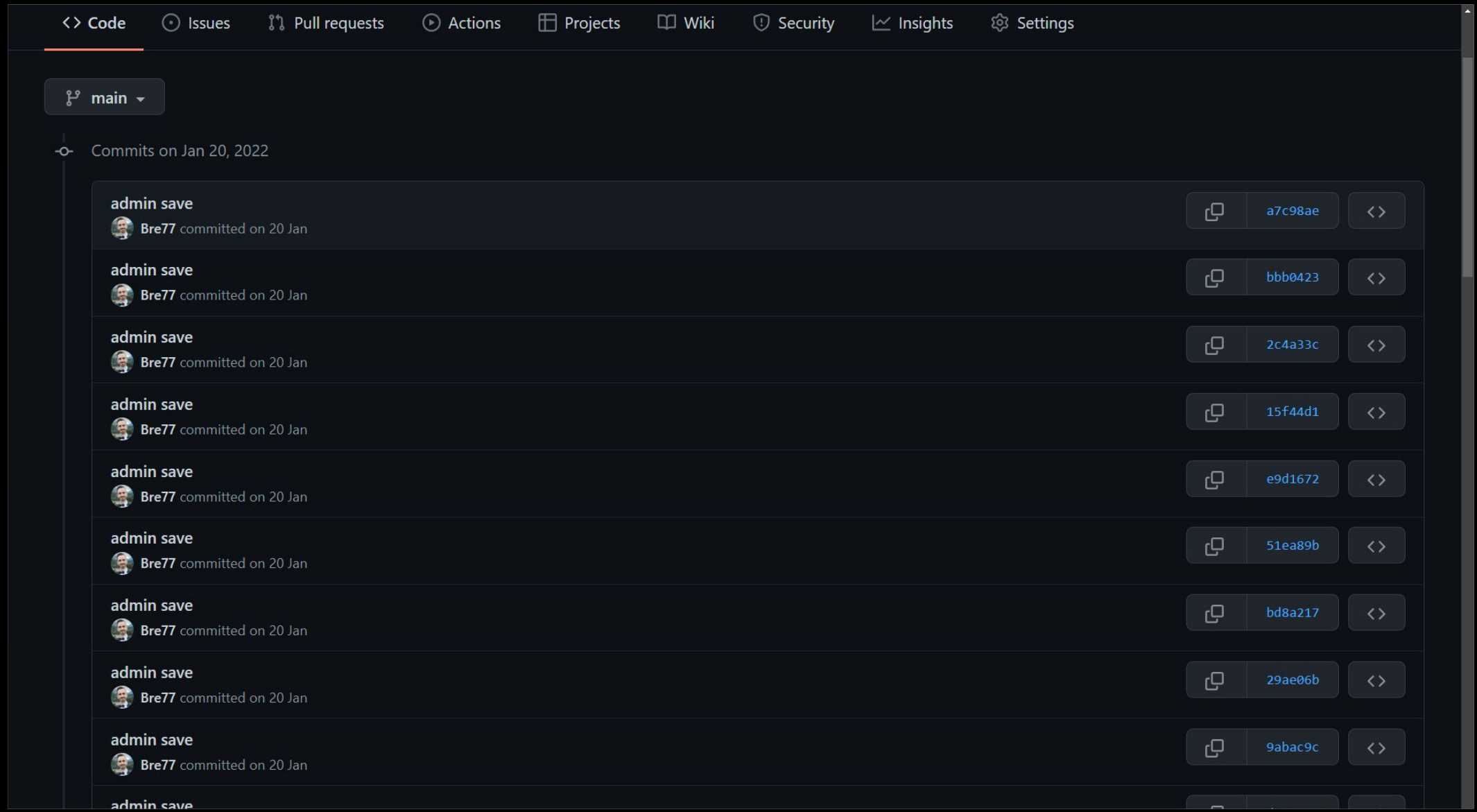
local/
local.meta
__pycache__
*.pyc

splunk> .conf22







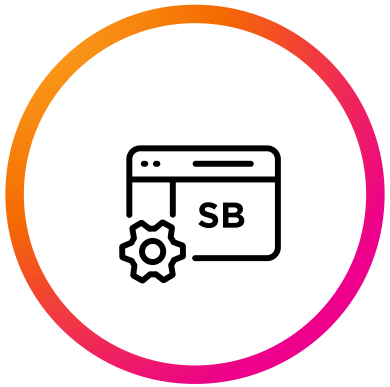


Recap

Create and Use Hooks

Speed up repetitive tasks

Reference the Spec & Btool



Avoid simple mistakes and write perfect conf

Test Fast and AppInspect

Using hooks, realtime search, and AppInspect

Easy Source Control

Publish your source (but not your passwords)

Git Hooked

On Config Explorer

https://github.com/Bre77/DEV1160B

Thank You

splunk> .conf22