Installation Guide for Leakage Detector

Requirements

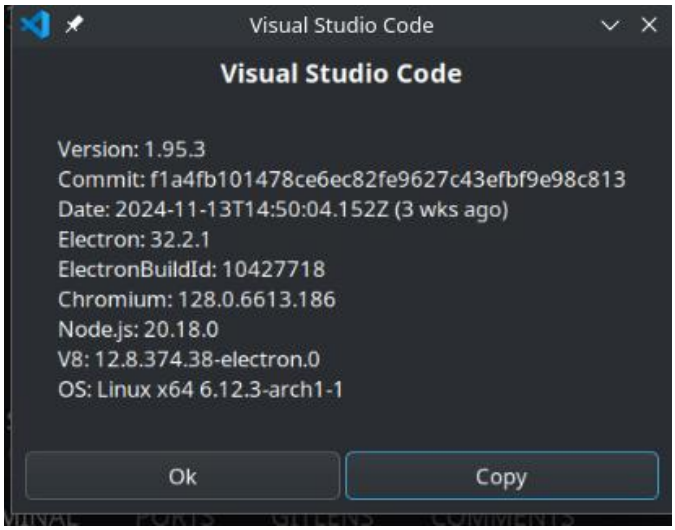
- Pre-built binaries for Windows, macOS, or Linux (failing that, Docker Engine or Desktop)
- NodeJS, which is required to run the prebuilt binaries for Windows, MacOS, and Linux
- Python extension for VS Code
- Jupyter Notebook extension for VS Code

Optional

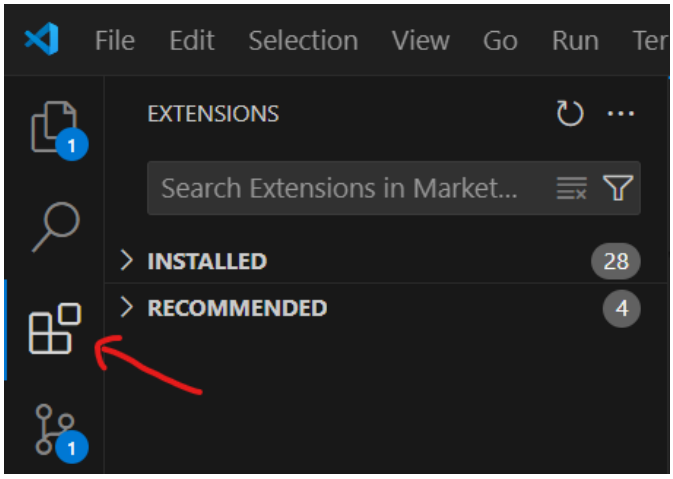
- GitHub Copilot, an AI-based VS Code extension for additional quickfix suggestions

1. Installing the VS Code Extension

1) Open Visual Studio Code. Here is one working version of VS Code for our extension (version 1.95.3):

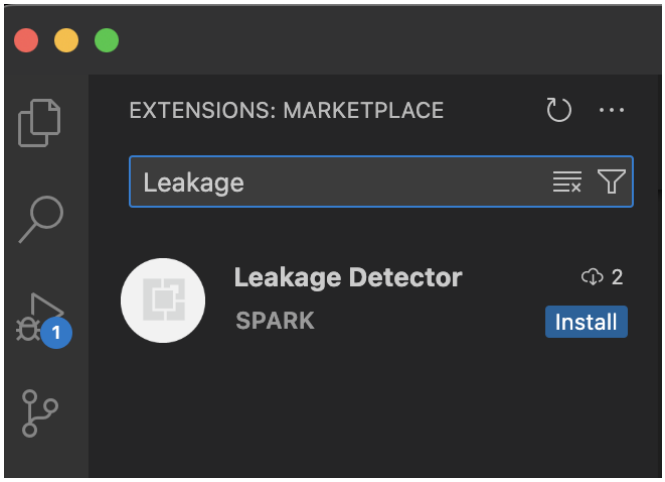


2) Navigate to the Extensions Marketplace by clicking on the Extensions icon on the Activity Bar or pressing Ctrl+Shift+X.



3) Search for 'Leakage Detector' in the search bar.

4) Look for the top result under the publisher 'SPARK' and click on it.



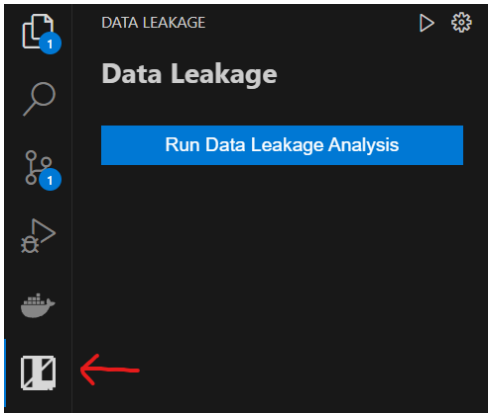
5) Click the 'Install' button to add the extension to your VS Code environment.

2. Downloading the Native Binaries

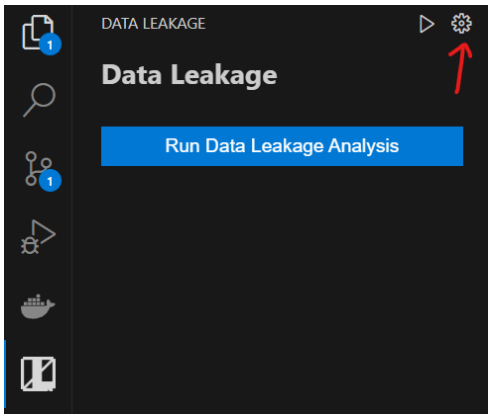
Disclaimer: if you only intend to use our Docker run mode solution, then downloading the native binaries is **unnecessary**.

Download the native binary through the extension in VS Code through these steps.

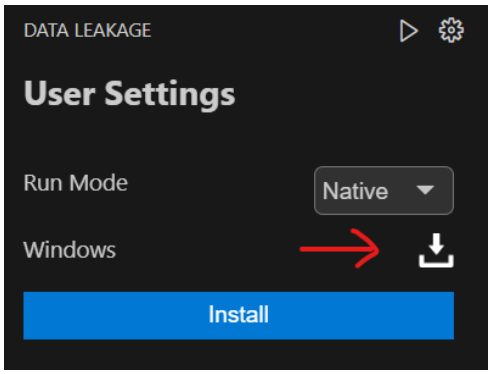
Step 1: Launch Visual Studio Code. From the activity bar on the left, choose the "Data Leakage" extension.



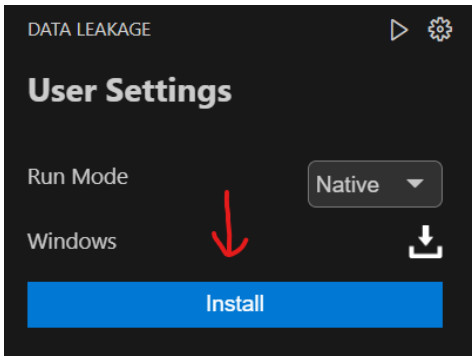
Step 2: Click on the settings icon to adjust the run settings.



Step 3: Click the download icon beside your OS to get the binary.



Step 4: If you downloaded the binary, extract this zipped binary to a directory of your choice. Click "Install" and select the extracted binary from your file directory.



The native binary is now ready for use with the Leakage Detector VS Code extension. Read our extension run guide or watch our tutorial video for further instructions on usage.

If you have **trouble installing** the binary, you can manually download it in the download page:

1) Go to the following URL in your browser: https://leakage-detector.vercel.app/download.
2) Identify and download the native binary that is most suited to your operating system (e.g., Windows, macOS, or Linux).
3) After downloading, locate the file and unzip it to a directory of your choice.
4) Once unzipped, the native binary is ready for use with the Leakage Detector VS Code extension.
5) Read our extension run guide or watch our tutorial video for further instructions on usage.

3. Docker Setup

Disclaimer: if you intend to use our native binaries with our VS Code extension, then this Docker setup is **unnecessary**. Otherwise, proceed to the next page for Docker setup instructions.

4. Resolving Antivirus/Firewall Issues on Windows

Windows OS users may encounter their antivirus or firewall automatically removing the native binary after downloading. Here is a solution to resolve these issues:

1) Open Windows Security by searching for it on the Start Menu.
2) Click on 'Virus & threat protection' in the menu.
3) Under the 'Virus & threat protection settings,' click on 'Manage settings.'
4) Scroll down to 'Exclusions' and click on 'Add or remove exclusions.'
5) Click on 'Add an exclusion' and select 'File' or 'Folder' based on where your native binary is located.
6) Navigate to the location of the downloaded native binary and select it.
7) Confirm the action and ensure the file or folder is now listed under exclusions.

5. (Optional) GitHub Copilot Setup

Our extension provides a naive manual Quick Fix option to fix data leakage. However, consider installing the GitHub Copilot extension in VS Code to enhance your data leakage fixes with AI-powered solutions. This tool can provide advanced code suggestions and fixes, including potential resolutions for data leakage issues in your Jupyter Notebook.

Steps to Install GitHub Copilot in VS Code:

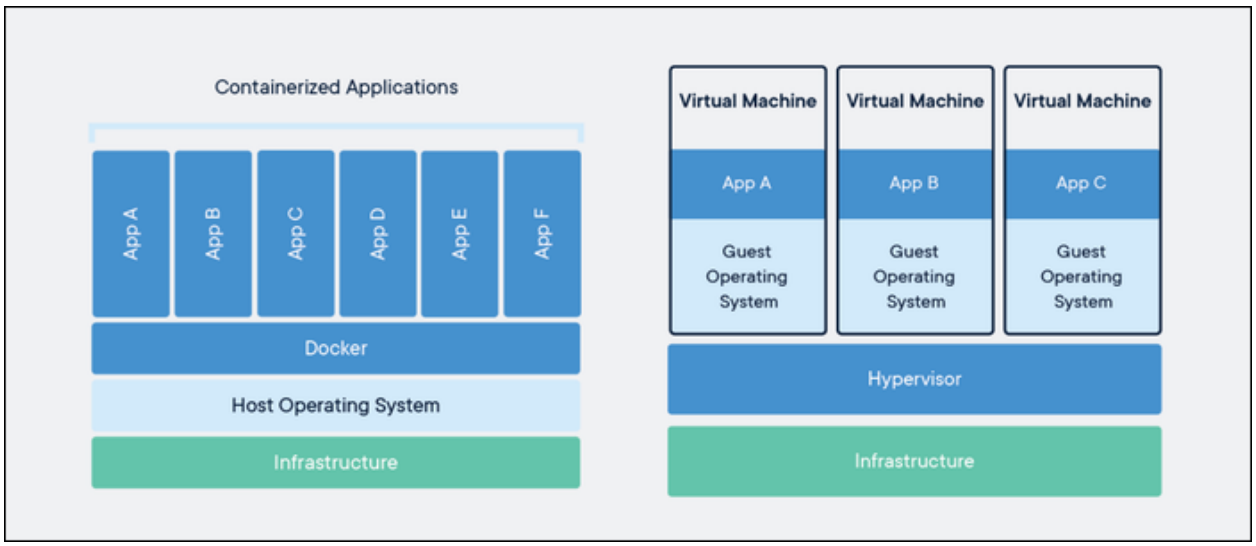
1) Open VS Code: Launch the Visual Studio Code application.
2) Access Extensions: Click on the Extensions icon in the Activity Bar on the side of the window.
3) Search for GitHub Copilot: In the Extensions view, type "GitHub Copilot" into the search bar.
4) Install Extension: Locate the GitHub Copilot extension in the search results and click the "Install" button.
5) Sign In to GitHub: Once installed, you may need to sign in with your GitHub account to activate the extension.
6) Enable GitHub Copilot: Follow the prompts to enable GitHub Copilot in your editor.
7) Start Coding: Open a Jupyter Notebook file and start using GitHub Copilot for AI-based code suggestions. Refer to our extension run guide to learn more about using GitHub Copilot's Quick Fix or our extension's manual Quick Fix to resolve data leakage.

Getting Started with Docker

A Guide to What Docker is & How to Install it

Introduction

So what is Docker? You can think of Docker as a middleman between full on virtualization and running something normally on your hardware. While virtualization would emulate an entire operating system on your computer, Docker uses something called containers. Containers, unlike VMs, use the host operating system and virtualizes the software running.



Containers only have exactly what they need inside of them, with nothing else. This means they are generally super lightweight so you can run a whole bunch of them on one machine. Instead of needing an entire desktop/virtual machine to run a simple application, like a web server for example, you can just throw it in a container and call it a day.

In a VM you call what manages the VM's the hypervisor (here's more information on hypervisors if you want to learn more), in Docker that's the Docker Engine.

Installing Docker

Docker is available for Linux, MacOS, and Windows, making it something very useful for developers who run into the “but it works on my machine” issue. It can also allow you to create a container, and anyone can run it on nearly any device without caring about the operating system. We will be using a package manager for all the installations. If you want to learn more about them/need to install one on your system, refer to the document Package Managers.

Windows Installation

There are many ways to install Docker on Windows, but we will be using Chocolatey, a package manager for Windows.

In an administrator PowerShell console, run the command. There may be a confirmation request in the console, confirm it or the installation will halt.

choco install docker-desktop

MacOS Installation

We are going to be using Homebrew for this, so install it if you haven't already. All you need to do is run the following command in your terminal if you have Homebrew installed.

brew install --cask docker

Running the Docker Desktop

For Windows and MacOS you need to run the Docker Desktop application that was installed on your local computer during the above steps. You need to do this to start the Docker Engine. Go through the recommended setup options (there is no need to create an account right now, skip where you can).

Verify your Install

To verify your install, run the command “docker run hello-world” and if you get an error message, you have not properly installed Docker.

Note: You will most likely need to restart your CLI to use the command after installation.

Docker Hub Account

Now that you have verified your installation, you must create a Docker Hub account or be logged in to Docker Hub. Think of Docker Hub as GitHub for Docker. To create a Docker Hub account, run “docker login” and follow the steps there.

Note: It is best to use a PAT (Personal Access Token)

Images

A Docker image describes the parameters of the container and how to create it. We don't always have to create our own images because there are a lot of them already available out there for us to use whenever we want! You can find a very large collection of them at the Docker Hub. Our VS Code extension handles the work of installing the Docker image and creating the container, so don’t worry about this. All you need to do is log in to Docker Hub and have Docker Desktop running in the background while the VS Code extension is running!