Code Dx Proof of Concept (PoC) Suggestions

Vincent M. Hopson – Field Applications Engineer at Code Dx

Here are a few items that can generally be used to create a compelling Proof of Concept at your site. All items listed here should be considered to be optional. We want to provide a framework to impart as much relevant knowledge about Code Dx as possible.

1 Static Analysis Tools (SAST)

Code Dx recommends that at least one commercial tool be used for your SAST testing. A big part of using our solution is the correlation between different tools. By using a commercial tool, and Code Dx's internal orchestration, you will be able to see findings that have been detected by multiple tools. This reduces the possibility of False Positives considerably.

2 Dynamic Analysis (DAST)

Normally, some form of dynamic testing is performed regardless of the application. Code Dx can receive the test results directly from tools like Portswigger's Burpsuite Professional, or the OWASP Zed Attack Proxy or ZAP. The latter is free, and the former is very low cost (at the time of this writing, about $300 USD).

We support plug-ins for both tools, and can take results from the tools in other formats in case you have unalterable copies.

It is possible to get results from other tools into Code Dx, and we would be happy to help get some of those into our product. This exercise helps with later automation needs you may have.

3 Developer Desktop Integrations

It is essential that ease of use be addressed in almost any setting. Especially if working with developers to address various findings. Code Dx supports integrations into the most popular developer environments:

- Microsoft™ Visual Studio
- Eclipse
- IntelliJ

Installation of the Code Dx integrations helps demonstrate how easy it is to gather results directly on the developer's desktop. By the security Engineer marking findings in Code Dx to indicate the person responsible, an easy search can create a list of issues to be addressed in the desktop environment.

4 Continuous Integration (CI) Installation

Code Dx supports many of the current Continuous Integration systems that are in common use. By taking a moment to install the plug-in, the CI environment can use Code Dx to determine the health of a build. This step could be used to provide the builder important security information, and automatically "break the build"; preventing potentially vulnerable code a path to your customers.

5 Dashboards

This tends to be more of a long term item as dashboard updating is performed once a day (usually early morning server time). By organizing the projects into sub-projects, it is possible to understand your security posture at a glance.

Data for the dashboards in Code Dx is available on a per-project basis. A switch in the top right corner of any dashboard display controls the available data. When sub-projects are populated with findings, the switch enables statistics from the current project to be displayed in isolation. Toggling the switch takes all of the available data from the current project and all sub-projects and displays it.

A valuable tool for assessing your progress in finding reduction and mitigation.

6 Report Generation or Custom Reports

Data accumulation is a tough task, but reporting on what you find effectively can be nearly as big a challenge. Information found inside of Code Dx can be used in a myriad of ways to help protect your code, or potentially your reputation outside of your organization. Reports can be generated inside of the tool as a number of formats with PDF probably being the most prevalent.

Using the Apache Format Object Processor (FOP) inside of Code Dx, we can generate alternate reports that cover the entire spectrum of intended audiences. From a risk assessment report to the CEO to a code block to be given to an Engineer. We can help you build effective reports that are black and white, or colorful descriptions of your security and/or quality stance.

7 Connect to Your Engineers

A couple of tight integrations help you to connect appropriate data to your Engineers. Using Git to read your repository and collect the most recent code is perhaps the most obvious. Any Git based system can be addressed.

Another common integration tool is to track various findings inside of Atlassian's Jira. Code Dx can be configured to push data to Jira on command, or automatically push tickets into Jira when analyses complete. This is a great way to show value by integrating Jira into the PoC workflow.

8 Penetration Test (PEN) Results

If you have results you can use from your PEN Testing in an electronic format, we can create a script (Python is my preferred scripting language, but we can use the language of your choice) to put those results into Code Dx. This operation will show you how to gather the results into Code Dx and what should be captured for correlations.

Note that an interface inside of Code Dx will allow entry of PEN test results, and is a good display of the information that would be needed for good correlation.

9 Threat Modeling

It is possible to ingest information into Code Dx from any phase of your development pipeline. Microsoft™ provides a free threat modeler that can be used to create results that Code Dx can ingest directly. This tool only requires a knowledge of your application (or a hypothetical one) to create useful information.

10 Hybrid Analysis

If your team is using Java or C#, it is possible to configure Code Dx to perform a deeper examination of your testing results. This “hybrid“ analysis can correlate SAST findings with your DAST operations.

There are two brands of *hybrid* analysis; trace based analysis (Java only), and traceless (both Java and C#). Setting these up is relatively easy, and can provide a much deeper window into how static findings relate to detected findings. These tend to have a *very low* False Positive rate.