Title: Windows Source-to-Sink Analysis Proposal

Problem Statement

Traditional methods of performing static analysis of binaries to identify vulnerabilities is often unscalable, leaving an analyst with the daunting task of examining thousands of binaries in order to triage them for vulnerabilities. A popular class of bugs that vulnerability analysts enumerate for are buffer overflows and command injections. Identifying these classes of bugs can often feel like looking for a needle in a haystack and is often time consuming. There have been recent advances toward scalable automated techniques that enable static analysis of binaries to discover command injections and buffer overflows by using data-flow-analysis. The capability has been successfully applied to discover vulnerabilities in embedded Linux devices.

Automation techniques have helped streamline the discovery of these classes of bugs. However, these tools often focus on the discovery of vulnerable functions on Linux binaries and have no utility on other operation systems such as Windows, as they have totally different API calls.

Proposed Approach

Begin by gaining a basic understanding of command injection vulnerabilities and what forms they often take. This can be done via Linux binaries, generating own examples would be useful, as well as discovering tools that provide users with potential command injection vulnerabilities. Then begin researching Windows APIs in order to understand how sources and sinks are utilized by developers. Again, creating their own examples may be useful. Additionally, researching vulnerable Windows binaries may provide guidance. Creating a curated list of Windows sources and sinks will be the end goal.

If time permits, students can attempt to integrate these findings into existing automated tooling.

Areas of Research: Windows security, Windows applications, data flow analysis, program vulnerabilities

Technologies Involved:

- Mango
- Binary analysis tools
- Control flow/data flow analyses
- Pwn.college training
- Static analysis tools
- Windows API

Student Participant Background Needed

This project focuses on computer security and common vulnerabilities. Students should have some familiarity with the following concepts:

- Command injections
- Binary analysis and reverse engineering
- Understanding of data flow and path analysis

Resources

Literature and other resources available for pre-project preparation:

- Extensive literature on operating systems, vulnerability and common security vulnerabilities
- Overviews and introductory materials for Windows and Unix API calls,
- Tutorials about how to perform static analysis, use binary analysis tools and perform vulnerability research/bug hunting
- Tutorials/knowledge of writing programs for Linux and Windows
- Mango Github: https://github.com/sefcom/operation-mango-public

Potential Cybersecurity Benefit

Windows commands 85% market share of desktop operating systems – having tools and analyses that can help Windows research out would be very beneficial. A lot of academic research efforts are focused on UNIX based operating systems, so this would provide a good baseline understanding for future knowledge.

Specific Tasks and Research Questions

- Given a list of vulnerability categories, what Windows functions are relevant in finding said vulnerabilities?
- Are there any differences in Windows binaries that would cause pre-existing source-sink analysis methodologies to fail?
- Are there any novel analysis shortcuts we can take given the differences in a Windows binary vs. UNIX binaries?

Open Issues

- Access to Linux environments for conducting research and generating tests
- Access to Windows environments and requisite tools for compiling Windows executables (e.g. Visual Studio)
- Access to materials (e.g. books) on Windows internals