|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Title:** | ***BugSpec - a standardized benchmark suite for security tools*** | | |
| **Date:** | **January 9, 2017** | | |
| **Researcher Name(s):** | | | **Daniel Krutz** |
| **University:** | | **Rochester Institute of Technology** | |

**New Project Proposal**

|  |
| --- |
| **Long Term Goal(s):** |
|  |
| Our goal with BugSpec is to have a widely accessible, widely applicable, and comprehensive test suite for measuring the efficacy of software analysis tools. |

|  |
| --- |
| **Background for Long Term Goal(s):** |
|  |
| Automated software testing systems have been an area of active research and development since the 1970s; however, due to the complexity of the software under test and the limitations of computer hardware, these systems haven’t been able to comprehensively measure software security or software quality. To deal with the complexity and hardware limitations, testing software has had to use approximations and heuristics, which leads to imprecision. We need to measure this imprecision to understand the gaps in analysis to further develop better software.  To measure the efficacy of software analysis tools, we need comprehensive benchmarks. Trail of Bits has been developing BugSpec for exactly this purpose. The BugSpec suite consists of over 240 C and C++ programs containing exploitable software vulnerabilities -- originally developed for DARPA’s Cyber Grand Challenge -- with input generators, sample crashing inputs, and corresponding patches. The BugSpec suite mimics real world software with real world vulnerabilities and was designed to measure the bounds of existing software analysis tools. |

|  |
| --- |
| **Intermediate Term Objectives:** |
|  |
| Our goal is to create a widely-applicable test suite that will be usable to researchers and engineers regardless of their chosen platform or method of analysis. To support this, we will take the DARPA Cyber Grand Challenge sample set of “challenge binaries”, which support only the custom DECREE operating system, and port them to commodity systems.  Our technical goals for BugSpec are:  • Support for Windows, Linux, and macOS platforms  • Support for 32bit and 64bit systems  • Unified cross-platform build system  • Cross platform input generators and support scripts  • Comprehensive patches (fixing unintended vulnerabilities as well) |

|  |
| --- |
| **Schedule of Major Steps:** |
|  |
| Task 1: Add support for Windows to BugSpec (6 weeks) The BugSpec benchmark suite already supports Linux and macOS. Our first major step will add full support for running the benchmark suite on Windows.  Task 2: Make the build system, input generators, and support scripts cross-platform (3 weeks)  Second tools and support software that BugSpec depends on must also work on Windows. This includes tools to compile individual benchmarks, generate test vectors for them, evaluate the test vectors, and collect testing results.  Task 3: Port all the benchmarks to work on 64-bit systems (6 weeks)  BugSpec was initially created for the 32-bit DECREE operating system and some individual benchmarks do not function correctly in a 64-bit environment. This task will find and fix these failure cases in BugSpec.  Task 4: Fix all failing benchmarks on Linux and macOS (3 weeks)  The majority of existing BugSpec benchmarks already work on Linux and macOS, however, a small few use specialized functionality that requires additional effort to port away from DECREE. This task will fix those remaining issues.  Task 5: Identify and fix unintended vulnerabilities in the benchmark suite (6 week)  There should be no unknown vulnerabilities in any BugSpec benchmark program, however, humans are fallable and unintentional vulnerabilities may exist. This task will validate that BugSpec contains only intentional vulnerabilities. |

|  |
| --- |
| **Dependencies:** |
|  |
| There are no major dependencies for further development of BugSpec. |

|  |
| --- |
| **Major Risks:** |
|  |
| BugSpec tests currently rely on POSIX system calls for much of their functionality. There is a risk that Windows support is more difficult than expected or is not possible due to this reliance on POSIX system calls. For example, there is no simple alternative to the select() system call on Windows and we will need to create a workaround.  Each vulnerable program in BugSpec includes a set of sample exploits, called Proofs of Vulnerability or PoVs. There is a risk that the existing PoVs, which work on Linux and macOS, do not work on Windows or on 64-bit operating systems. This would necessitate a rewrite of the existing set of PoVs and could add considerable time to our schedule. |

|  |
| --- |
| **Budget:** |
|  |
| Our budget for BugSpec is $50,000. |

|  |
| --- |
| **Staffing:** |
|  |
| BugSpec will be staffed by two engineers from Trail of Bits.  Kareem El-Faramawi, a Security Engineer, will perform the majority of the research and development activities to meet the intermediate goals of the project. Kareem has been the primary maintainer of BugSpec since it was originally created.  Ryan Stortz, a Principal Security Engineer, will provide oversight, direction, and technical advisory to the project, including such tasks are code reviews. Ryan has spearheaded the technical direction of the BugSpec project since its inception. |

|  |
| --- |
| **Category of Current Stage:** |
|  |
|  |

|  |
| --- |
| **Contacts with Affiliates:** |
|  |
| Kevin E. Greene, DHS  kevin.greene@hq.dhs.gov |

|  |
| --- |
| **Publications and Other Research Products:** |
|  |
| Zlib Security Audit  Trail of Bits performed an automated security audit of zlib for the non-profit Mozilla Foundation, the maintainers of the Firefox web browser and Thunderbird email client. Zlib is an open-source compression library with a history of security vulnerabilities. Trail of Bits used the automated vulnerability discovery tools developed for the Cyber Grand Challenge to automatically audit zlib for security vulnerabilities, which delivered confidence while reducing costs to a level acceptable by a non-profit.  Osquery  Facebook’s osquery is an open-source endpoint security tool that allows an organization to treat its infrastructure as a database. Using osquery, it is possible to issue SQL queries about organization state (e.g. list every process that is listening on port 445 across the entire network infrastructure). Originally, osquery only supported Linux and macOS, and Facebook contracted Trail of Bits to port the osquery agent to Windows.  DARPA Cyber Grand Challenge  We competed as one of 7 funded teams in DARPA’s Cyber Grand Challenge. As part of the competition, we’re developing static and dynamic analysis tools to automatically detect, mitigate, and exploit software vulnerabilities. These tools are now used for performing automated audits of commercial software.  McSema x86 to LLVM IR binary lifter  We developed and open-sourced mcsema, a framework that performs static translation of x86 binaries to the LLVM intermediate representation. McSema allows us to leverage LLVM’s comprehensive analysis platform on binary code for software transformation, vulnerability discovery, software optimization, and other uses. This framework has been adapted and is used by several DARPA programs. The code is available at https://github.com/trailofbits/mcsema.  6.3 DARPA Cyber Fast Track - Code Reason  Code Reason is a research project to build a framework for symbolic reasoning about code and semantic discovery of code snippets that can be utilized in return-oriented-programming (ROP) attacks.  6.4 DARPA Cyber Fast Track - PointsTo  PointsTo is a research project that is intended to find object life-cycle (use-after-free, use-after-return) vulnerabilities in large software projects such as web browsers and web servers.  For further information about the research activities of Trail of Bits, please see our research blog:  https://blog.trailofbits.com/2017/01/09/2016-year-in-review/ |

|  |
| --- |
| **References:** |
|  |
| Dmitri Alperovitch <dmitri@crowdstrike.com>  CrowdStrike, CTO  Senior Fellow, Atlantic Council  Vice President, Threat Research, McAfee (former)  Byron Cook <byron@amazon.com>  Senior Principal, Amazon  Professor, University College London Researcher (former)  Researcher, Microsoft Research (former)  Mike Arpaia <mike@arpaia.co>  CTO/co-founder, Kolide  Lead Developer of osquery (https://osquery.io/)  Engineering Manager, Facebook (former)  Ryan Perry <ry@junelife.com>  Director of Software, June Life  Senior Engineering Manager, Twitter (former)  iPhone Software Engineer, Apple (former) |