USENIX

THE ADVANCED COMPUTING SYSTEMS ASSOCIATION

“I’m trying to learn. . . and I’m shooting myself in the foot”: Beginners’ Struggles When Solving Binary Exploitation Exercises

James Mattei, Christopher Pellegrini, and Matthew Soto, *Tufts University;* Marina Sanusi Bohuk, *MetaCTF;* Daniel Votipka, *Tufts University*

https://www.usenix.org/conference/usenixsecurity25/presentation/mattei

This artifact appendix is included in the Artifact Appendices to the Proceedings of the 34th USENIX Security Symposium and appends to the paper of the same name that appears in the Proceedings of the 34th USENIX Security Symposium.

August 13–15, 2025 • Seattle, WA, USA

978-1-939133-52-6

Open access to the Artifact Appendices to the Proceedings of the 34th USENIX Security Symposium is sponsored by USENIX.

USENIX Security '25 Artifact Appendix "I'm trying to learn... and I'm shooting myself in the foot": Beginners' Struggles When Solving Binary Exploitation Exercises

James Mattei*, Christopher Pellegrini*, Matthew Soto*
Marina Sanusi Bohuk†, and Daniel Votipka*
*Tufts University; †MetaCTF

A Artifact Appendix

Welcome to our artifact. There are various files provided in the OSF repository; the three that are directly needed for evaluating functionality and reproducibility are Artifact.zip, CleanData.zip, and Artifact.py.

A.1 Abstract

[Mandatory] Vulnerability discovery is an essential security skill that is daunting for beginners to develop as there is a large body of knowledge to learn. They are often encouraged to search the web for information or "Just Google It!", but whether this advice is effective is unclear. We conducted semi-structured observational interviews with 37 vulnerability discovery beginners attempting to exploit 51 vulnerable programs. We capture the questions beginners have when trying to identify and exploit vulnerabilities, how they search for answers, and the challenges they face applying the results of their searches. We performed a rigorous qualitative coding of our dataset of 3950 events characterizing our participants' actions to identify several behaviors and obstacles they faced, along with quantitative measures to determine their most frequent issues.

We found beginners struggle to understand how to exploit vulnerabilities, craft their solutions, and even complete common technical tasks. They were often unable to find relevant information online to overcome these struggles, as they lacked the relevant vocabulary to craft effective keyword searches. When they did find relevant web pages, they struggled to properly transfer information from the web to their challenges due to misunderstandings and missing context. Our artifact contains the raw data for each participant run, along with the code to produce the key statistics and figures in the paper. The class definitions and data model for our participant data are flexible and allow for further analysis or use of the data set. Please take note of the chi-squared statistical tests, the generated line graph, and the counts of different events.

A.2 Description & Requirements

[Mandatory] To properly run the artifact you will need to extract both CleanData.zip and Artifact.zip, placing those folders at the same location as Artifact.py. You will need a valid installation of Python 3 and approximately 2 MB of storage for the data.

A.2.1 Security, privacy, and ethical concerns

[Mandatory] There should be minimal to no security, privacy, or ethical concerns for evaluators of our artifact. The data in CleanData.zip is already anonymized and only has the timestamps of our applied codebook. It does not contain any raw interview videos or materials that could be used to deidentify participants. Similarly, our artifact is not inherently destructive, so there are no risks to running Artifact.py.

A.2.2 How to access

[Mandatory] All artifact components can be found at the following OSF repository `https://osf.io/y6xew/`. Not all files at this repo need to be downloaded for evaluating the functionality and reproducability; only Artifact.zip, CleanData.zip, and Artifact.py are needed to run the artifact properly.

A.2.3 Hardware dependencies

[Mandatory] None

A.2.4 Software dependencies

[Mandatory] None

A.2.5 Benchmarks

[Mandatory] None

A.3 Set-up

[Mandatory]

A.3.1 Installation

[Mandatory] First, you must extract the two zip files Artifact.zip and CleanData.zip so that their contents are stored at the same location as Artifact.py. Next, you must activate the Artifact Python virtual environment, which you can do with the command:

source Artifact/bin/activate

Lastly, you can run the artifact program with the command: *python3 Artifact.py*

A.3.2 Basic Test

[Mandatory] There is no real way to run a basic test without running the entire program. But on the first time running the program, it may take up to one full minute before you start to see output. Once you start seeing lines appear stating *Building: [DATAFILE]*, then you know the artifact is running properly and able to read in the participant data properly.

A.4 Evaluation workflow

[Mandatory] After running the artifact, there are two locations with output that should be validated: the console where you ran the program, and a newly created output folder called "ArtifactOutput. The Code for generating the figure begins on line 1215."

A.4.1 Major Claims

[Mandatory for Artifacts Functional & Results Reproduced, optional for Artifact Available]

(C1): *Figure 2 showing the number of participants with different counts of Mistakes and Web Pages by problem step. This output is reproduced in "CodeTypeByStep.png" in the ArtifactOutput folder.*

(C2): *Table 3 showing the counts of different events broken down by which step they occurred during. This output is given in the file CountByStep.csv generated in the ArtifactOutput folder, and the code for calculating it begins on line 1307.*

(C3): *Throughout the results, there are references to Chi-squared test results. The full table of those tests are shown in the supplemental PDF aptly named "supplemental.pdf" in the same artifact OSF folder. These numbers are displayed on the console and the code for calculating them begins on line 1086.*

In addition to the data related to the major claims, there is a folder generated in the "ArtifactOutput" folder, which contains additional descriptive figures about participant interviews that were used for axial coding but ultimately did not explicitly make it into the final paper. As an example, there is one graph displayed that visualizes the actions a participant took through the duration of a given step. For more information, check the README.txt file provided in the OSF repository.

A.4.2 Experiments

[Mandatory for Artifacts Functional & Results Reproduced, optional for Artifact Available]

(E1): *Statistical results from analysis.* There are no specific experiments to run for our paper per say, but all of the major claims can be validated within the one provided script. It should take no more than 10 human-minutes to properly set up the environment, and five compute minutes to process the data. Please follow the instructions in this document and the README in the artifact folder for details about running the file.

A.5 Notes on Reusability

[Optional] Our artifact shows a potential first pass at our rich data set to extract insights. Others could reuse this artifact to investigate additional facets of beginner struggles. The data structures and data processing defined in the first 1000 lines is a starting point for investigating more detailed questions or replicating this study with our codebook and coding practices in a new environment.

A.6 Version

Based on the LaTeX template for Artifact Evaluation V20231005. Submission, reviewing and badging methodology followed for the evaluation of this artifact can be found at https://secartifacts.github.io/usenixsec2025/.