Reproducibility in Applied Security Conferences: An 11-Year Review on Artifacts and Evaluation Committees

Daniel Olszewski
University of Florida
Gainesville, Florida, USA

Allison Lu
University of Florida
Gainesville, Florida, USA

Anna Crowder
University of Florida
Gainesville, Florida, USA

Nathaniel Bennett
University of Florida
Gainesville, Florida, USA

Seth Layton
University of Florida
Gainesville, Florida, USA

Sri Hrushikesh Varma Bhupathiraju
University of Florida
Gainesville, Florida, USA

Tyler Tucker
University of Florida
Gainesville, Florida, USA

Siddhant Kalgutkar
University of Florida
Gainesville, Florida, USA

Hunter Ver Helst
University of Florida
Gainesville, Florida, USA

Carson Stillman
University of Florida
Gainesville, Florida, USA

Sara Rampazzi
University of Florida
Gainesville, Florida, USA

Kevin R. B. Butler
University of Florida
Gainesville, Florida, USA

Patrick Traynor
University of Florida
Gainesville, Florida, USA

Abstract

Reproducible research ensures a foundation for credible and progressive scientific advancement. Calls for reproducibility in the computer security and privacy community resulted in ACM WiSec and ACSAC in 2017 creating artifact evaluation committees (AECs) to foster more reproducible outcomes. Upon acceptance to ACM WiSec and ACSAC, authors may submit experimental artifacts to the committee to demonstrate that their artifacts effectively support their paper. While this may seem to address reproducibility, it is unknown whether AECs have actually improved reproducibility. Furthermore, AECs only assess artifacts submitted to the committee, and thus, may not holistically reflect the status of reproducibility (e.g., artifacts that did not go through the AEC). We conduct an indirect (i.e., availability) and a direct (i.e., running code) reproducibility studies to measure and evaluate the potential and realized reproducibility by classifying and running artifacts associated with individual papers. We collect over 2,000 papers and 550 artifacts to measure the state of reproducibility within four established applied security conferences between 2013 and 2023, two of which have AECs and two comparable conferences without AECs. We find that only 10% of research provides runnable artifacts, and only 3.9% of published papers produce a reproducible artifact. More importantly, our work shows that including an AEC alone does not necessarily result in more reproducible artifacts; specifically, ACM WiSec artifacts are more or less in line with the two control conferences, whereas ACSAC ultimately achieves 90% of artifacts participating in the AEC, with 40% of artifacts running. As such, our work shows that while the security community is developing mechanisms to foster reproducible research, there are significant improvements needed to progress reproducibility within computer security and privacy research.

Keywords

Computational Reproducibility, Artifact Evaluation Committees, Applied Security Reproducibility, State-of-Practice

ACM Reference Format:
Daniel Olszewski, Allison Lu, Anna Crowder, Nathaniel Bennett, Seth Layton, Sri Hrushikesh Varma Bhupathiraju, Tyler Tucker, Siddhant Kalgutkar, Hunter Ver Helst, Carson Stillman, Sara Rampazzi, Kevin R. B. Butler, and Patrick Traynor. 2025. Reproducibility in Applied Security Conferences: An 11-Year Review on Artifacts and Evaluation Committees. In . ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 Introduction

Computational reproducibility acts as a cornerstone for validating findings and conducting reliable research. Reproducible research garners confidence in results, enables collaboration amongst teams, and advances subsequent research. In recognition of how important reproducibility is to the state of science, many influential institutions (e.g., the National Academies of Science, the National Science Foundation, and the Association for Computing Machinery) have produced guiding principles for fostering reproducible research (e.g., prioritize confirmatory research) and solicited responses to improving the state of reproducibility [6, 17, 18]. The nature of computer security research and the broader field of computer science involve the inherent development and construction of experimental artifacts, encompassing deployable code, data aggregation scripts, and analytical methodologies that define research outcomes. In response to the growing calls for reproducible research, conferences have instituted artifact evaluation committees (AECs) that evaluate artifacts from accepted manuscripts and determine whether the associated artifact reproduces the claims of the paper.

To measure the effectiveness of artifact evaluation committees, a single prior work [20] measured machine learning (ML) artifacts across notable security conferences to understand the impact of AECs on reproducibility. However, their findings did not include Association for Computing Machinery Conference on Security and Privacy in Wireless Security and Mobile Networks (ACM WiSec) and Annual Computer Security Applications Conference (ACSAC), even though both conferences created artifact evaluation committees *three years* in advance of the only conference in their study that had an AEC. Thus, our work considers the state of reproducible research amongst conferences in applied computer security, including ACSAC and WiSec, to provide a more comprehensive understanding of the evolution of reproducibility practices and the efficacy of artifact evaluation committees by expanding the scope and quantity of measured papers.

To characterize the state of computational reproducibility in applied computer security, we conduct a comprehensive evaluation of artifacts available at ACM WiSec and ACSAC and two comparable conferences, IEEE European Symposium on Security and Privacy (Euro S&P) and Association for Computing Machinery Asia Conference on Computer and Communication Security (AsiaCCS). Thus, we make the following contributions:

- **Reproducibility Study:** We measure the indirect and direct reproducibility of four premier applied security conferences (ACSAC, ACM AsiaCCS, Euro S&P, and ACM WiSec) over the past 11 years, two with AECs and two comparable conferences without AECs. In total, we consider over 2,000 papers, and find that 579 papers make their artifacts available and that the percentage of papers with artifacts has increased by 36% in the past ten years. We are able to compile 35% of the artifacts. We containerize every available artifact and measure the state of reproducible practices in applied security conferences.
- **Analyzing the Effect of AECs:** ACM WiSec and ACSAC introduced their AECs in 2017 to improve the state of artifacts at the conferences. We evaluate to what extent AECs have made an impact on the state of reproducibility, and show that artifact participation at ACSAC is statistically higher than WiSec, with over 90% of artifacts at ACSAC evaluated by the AEC. Further, we find that artifacts that went through ACSAC's AEC were statistically likely to run more often.
- **Characterizing Outside Factors:** Finally, we explore facets of reproducibility (e.g., programming language, topic area, and funding agency) to understand whether reproducible outcomes are affected by outside factors. We find that artifacts written in C compile more often than other languages and that the National Science Foundation funds the most projects with compilable and/or runnable code. Further, we discuss open challenges for reproducibility and what lessons outside communities can learn from the applied computer security and privacy community.

This work represents over four person-years of work with thousands of hours of computation. We are in the process of deploying all 579 artifacts to the SPHERE [2] project, a scalable initiative to provide computing resources suited to security researchers to promote reproducibility, to provide the artifacts of our research for future investigation. The remainder of this paper is organized as follows: we discuss relevant background and related work in Section 2; we summarize the methodology and results of our indirect and direct study in Section 3; Section 4 compares our work to prior work and analyzes machine learning reproducibility and artifact evaluation committees; we explore factors that affect reproducibility in Section 5; Section 6 provides recommendations and limitations; and finally, Section 7 concludes.

2 Background

2.1 Reproducibility

While many definitions exist of reproducibility and its variations (e.g., repeatability, replicability, and generalizability), we focus on the definition provided by the National Academies of Science [18]. Reproducibility is obtaining consistent results using the same data and code. In computer security, reproducibility often refers to recreating the same computational analysis; we refer to this as computational reproducibility. Reproducibility studies conducted by an independent team are done in two parts [18], indirect and direct. An indirect study classifies the plausibility of reproducibility by identifying the necessary artifacts to reproduce a study (e.g., code, data, methodological descriptions). A direct study attempts to run the associated artifacts and reproduce the results, often emulating the environment the code runs on. Direct reproducibility studies focus on recreating the results attained in the paper by using *the same code and data as in the original study.* In this work, we conduct both an indirect and direct study of reproducibility.

2.2 Artifact Evaluation Committees

With calls for reproducible research growing in the security community, conferences are implementing Artifact Evaluation Committees (AECs) with a call for artifacts. AECs encourage authors to provide reproducible artifacts, often for papers *that are accepted to the conference.* Submission to an AEC is not required but results in badges and awards given to papers that go through the AEC process. The three badges designated by the ACM are as follows: Artifacts Available, if the artifacts are made accessible online; Artifacts Functional, if the artifacts compile and/or run with minimal modification of the code; and Results Reproduced, if the paper's claims are reproduced by an independent team [6]. Artifacts Functional is characterized by four factors: *documented*, *consistent*, *complete*, and *exercisable*. It is important to note that AECs do not interact with reviewers for a paper, and as such, do not influence or provide feedback to the publication process.

AECs in the security community were pioneered by WiSec and ACSAC in 2017 [4, 5]. Both conferences detail reports about the

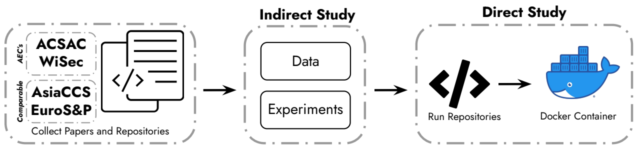

Figure 1: Reproducibility workflow of this work: we collect papers from ACSAC and WiSec that have AECs, and AsiaCCS and EuroS&P as comparable conferences. We perform the indirect study first, followed by the direct study.

artifacts submitted and provide links to each artifact. The first larger security conference to introduce an AEC was USENIX Security in 2020 [7]. In 2021, USENIX Security modified its AEC to align with ACM's badging and also published a report on the reproducible artifacts [8]. ACM CCS introduced an AEC in 2023 [9], but no badges or reports were made public. Thus, it is difficult to ascertain how many papers were submitted to this AEC. In our work, we consider both artifacts that have been evaluated by an AEC and artifacts that have not.

2.3 Reproducibility in Computer Security

While numerous areas of science are focusing on meta-science papers, to date, only two large-scale reproducibility studies exist within the security community. Olszewski et al. [20] conducted an indirect and direct reproducibility study of machine learning (ML) security papers. They looked at 750 ML papers at four security conferences (USENIX Security, ACM CCS, IEEE S&P, and NDSS) between 2013 and 2022 and found that only 40% of the studied papers include artifacts, and of the available artifacts, only 44% work (i.e., 17.6% of the 750). Schloegel et al. [24] surveyed over 150 fuzzing papers published between 2018 and 2023. They then attempted to reproduce eight of the papers but faced difficulties in reproducing all of the original authors' claims.

While these large studies created a foundation for reproducibility studies, both papers focus on granular areas within the security community. As such, their findings only address machine learning security and fuzzing and do not fully explore the relationship between AECs and reproducibility, given the limited existence of these committees compared to venues such as ACSAC and ACM WiSec. Further, applied security conferences introduced AECs before the notable conferences. We focus on the origins of AECs in security and comparable conferences.

3 Indirect and Direct Study

In this section, we present our adopted methodology for collecting papers and artifacts of applied security conferences (ACSAC, AsiaCCS, Euro S&P, and ACM WiSec) and our results for our indirect and direct studies. As stated previously, we select ACM WiSec and ACSAC as they were the first to introduce AECs to the security community. Further, we select two comparable conferences (e.g., ACM AsiaCCS and IEEE Euro S&P) without AECs as a control. Our methodology is visualized in Figure 1.

3.1 Methodology

We examine papers on systems, network, machine learning, cryptography, hardware, and mobile security from four applied security conferences (e.g., ACSAC, AsiaCCS, Euro S&P, and WiSec) ranging from the years 2013 through 2023, excluding posters and workshop papers. We collect 628 from ACSAC, 789 from AsiaCCS, 290 from EuroS&P, and 302 from WiSec for a total of 2,009 papers.

We select papers with any code or website that the authors created that are mentioned anywhere in the paper and are directly relevant to the authors' methodology. We consider all papers published at each conference, not just papers submitted to the AECs. Due to the large number of papers, each conference paper is reviewed by two reviewers to identify artifacts and code.

We use the following criteria in our selection process: (1) the authors include a link to their code or data;[1] (2) the code or data is directly relevant to the authors' findings. We define direct relevance to the paper as code or data that the authors created in order to carry out their experiments, but not tools or code used by the authors but not created by them (e.g., WireShark for network captures). After finishing our paper selection process, we found a total of 579 papers that satisfy these criteria.

3.1.1 Selection Criteria. We consider the availability of experimental artifacts in our coding process (i.e., the availability of code and instructions), ReadMe/directions, type of repository, programming language used, if the code compiles, hardware required to run the code (e.g., GPU, Raspberry Pi, etc.), and if the code was made for reproducibility. We classify our results in terms of the ACM's requirements for a functional artifact (e.g., documented, consistent, complete, and exercisable) [6]. A *documented* artifact requires an accounting of the contents of the artifact as well as instructions to run the artifact. *Consistent* (i.e., made for reproducibility) refers to the artifact in some way reflecting the experimental results. An artifact is made for reproducibility if the artifact directly runs experiments and achieves outcomes of the paper (e.g., if the paper claims a performance boost over a network protocol, the artifacts are made for reproducibility if the artifact runs a performance test and reports it). A *complete* artifact captures all experimental outcomes. In our work, we do not evaluate the completeness of an artifact, though we do make note of it when necessary. Finally, an artifact is *exercisable* if the code in the artifact compiles and/or runs.[2] This step is used to evaluate the efficacy of the available artifacts in recreating the results for each paper. We consider the quality of instructions, the availability and revision of code and data, whether or not the code runs, whether the output of the artifact matches the metric in the paper, and the consistency of our results with the paper's claims.

When given the link to the repository, we first download the repository to our servers. Some repositories and their associated data may require an inquiry for access, which we do so by getting in contact with the author's point of contact for the paper. If the artifact contains a Docker image, we run our analysis using that. Otherwise, we set up a Docker container and clone the repository into the Docker container. Following instructions (if provided) from the ReadMe or similar page, we spend a maximum of one hour to debug or set up the project. If the project does not run after the

[1] Usually found in a footnote or the Methodology section of the paper.
[2] We say compiles and runs for both compilable and interpreted code bases.

allotted hour, the project is not runnable in our experiment, similar to prior work [12, 20]. In the case of machine learning papers where training is required or a repository uses C/C++ code that needs large packages installed, we allow a maximum of 10 hours, similar to prior work [20, 22]. Some repositories may also require the use of certain hardware. In the case of GPUs, we attempt to match the GPU as best as possible, but we only have access to Nvidia A100s and GeForce RTX2080TIs. For all other hardware requirements, we use equipment that is already available to us if possible. Some require specialized hardware that we are unable to procure. In that case, we evaluate it in the indirect study but do not run the source code. Finally, if an artifact is made for reproducibility and runs, we consider an experiment reproducible if our results are within 10% of the claimed metrics, similar to Raff [22].

3.2 Indirect Study

An indirect study measures the availability of factors that enable reproducibility (e.g., artifacts or written methodologies). Although indirect studies represent the start of direct studies (i.e., a direct study requires identifying the presence of artifacts), an indirect study analyzes the presence of reproducible work within not only the artifacts but also the papers as well. As such, in this work, we explore the availability of code artifacts, instructions for running the artifacts, and the purpose of the artifacts.

Results. Of the 2,009 papers, 579 (28.9%) provide artifacts that are directly associated with the paper. As seen in Figure 2, the total for each conference is 230 (36%) for ACSAC, 125 (17%) for AsiaCCS, 84 (28%) for EuroS&P, and 69 (23%) for WiSec. When identifying the associated artifacts, we found 56 broken links to code repositories. Of the 56 broken links over 50% of the repositories were published in the past four years. While we obtained access to most artifacts that required requesting access, there were two remaining repositories. For one of them, the email to request access no longer exists.

Observations. When reviewing the documentation of the artifacts, we compare the ReadMe/instructions provided and look at the goal of the artifacts. A well-documented artifact includes a description of what is in the repository and instructions on downloading, installing, setting up, and running the artifact. For example, EF/CF by Rodler et al. [23] contains a detailed ReadMe with step-by-step instructions to run the Docker image and additional explanations for what each step accomplishes. We find that 54% of ACSAC, 18% of AsiaCCS, 31% of EuroS&P, and 45% of WiSec papers contain a complete ReadMe (e.g., have both a description of the artifact and instructions to run). We find that 8% of ACSAC, 14% of AsiaCCS, 10% of EuroS&P, and 13% of WiSec artifacts contained no instructions or description of the artifact (e.g., a blank ReadMe). In one case, the repository states "Instructions are coming soon", yet the repository was last edited in 2020. As mentioned earlier, a consistent artifact (i.e., made for reproducibility) recreates the experimental claims by outputting results that are shown in the associated paper. Thus, simply putting code online does not satisfy this requirement. For example, the code repository for *aaecaptcha* by Hossen et al. [16], which generates adversarial captchas, not only has code for the implementation but also contains code for evaluating the attacks against multiple automatic speech recognition systems. Thus, we consider this artifact made for reproducibility. If the repository

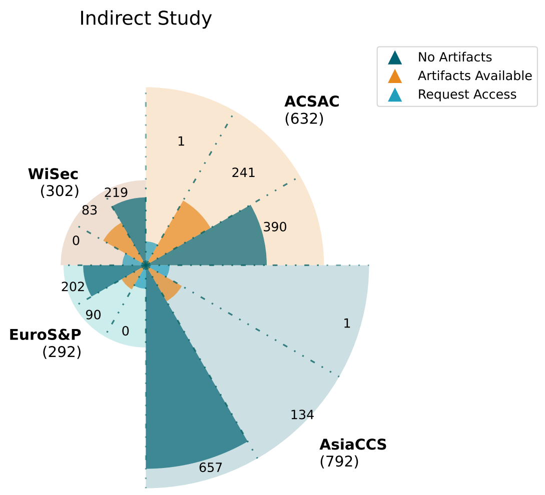

Figure 2: A Coxcomb graph of the results of our Indirect Study. We see that for ACSAC, AsiaCCS, EuroS&P, and WiSec, each conference contained 241, 134, 90, and 83 artifacts, respectively. While AsiaCCS has the most papers, ACSAC has by far the highest number and percentage of available artifacts. We use the public source code from Olszewski et al. [19] to generate our figures for easy comparison.

does not have the evaluation, we do not consider it made for reproducibility. Further, some repositories are packaged tools, a valuable resource for the community, that do not have experimental results. These are not made for reproducibility.[3]

We find that 48% of ACSAC, 26% of AsiaCCS, 31% of EuroS&P, and 38% of WiSec artifacts are made for reproducibility. We find some repositories only reproduce part of the claims. For example, the code may evaluate two of the seven statistical tests discussed in the corresponding paper, but does not provide anything further for the remaining five tests. As discussed in Section 6, we take the most positive aspect and count reproducing part of their results as reproduced. While often the instructions and code are in the linked repository, the code and the instructions are not always stored in the same directory or website, or unavailable (e.g., a broken link).

Takeaway. *Supplementary artifacts in computer security are sometimes designed for reproducibility, but there is a lack of maintenance on artifacts that negatively affects reproducibility.*

3.3 Direct Study

The direct study focuses on running the artifacts associated with each paper and determines the exercisability of an artifact. Figure 3 depicts the results of our direct study.

Results. We find that we can compile and run 35% of the artifacts; 22% of these artifacts ran with no edits or debugging, and 13% ran with minimal edits. 51% of the code repositories did not run after an hour of debugging. We are unable to run 57 (11%) of the

[3]We do not create scripts to reproduce the results from the tool, but we do attempt to compile the tool.

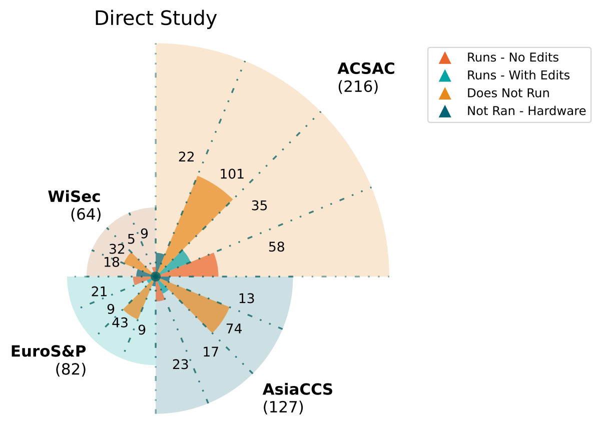

Figure 3: We show the success of running code in Coxcomb graph for each conference.

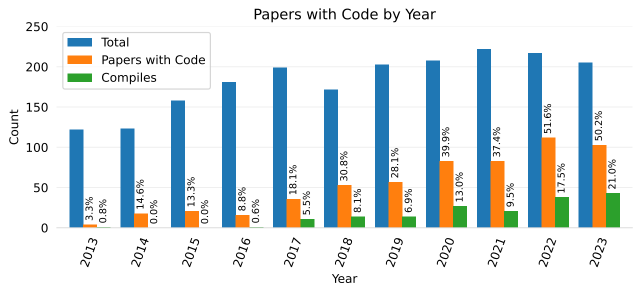

Figure 4: Count of code artifacts and compile rate by year.

artifacts as it requires external hardware or a license. 13 repositories (3%) exceed the 10-hour running limit (i.e., we could run them, but the program never terminated after 10 hours). Thus, only 10% of published research papers provide a compilable artifact. When we look at the number made for reproducibility and compilation, we find that 78 (3.9%) of published papers provide artifacts that compile for reproducibility. When comparing conferences, we find that the artifacts compiled for ACSAC at 40%, AsiaCCS at 29%, EuroS&P at 36%, and WiSec at 20%. Figure 4 shows the number of papers, the number of artifacts, and the number that compile per year. Overall, we notice that in 2023, the number of artifacts per paper is up to 50% compared to 3% from 2013. We see a growing trend of artifact inclusion within conferences. Furthermore, we see that the percentage of compiled artifacts is increasing over time as well. We confirm this by conducting a linear regression on both the number of artifacts available and the number of compiled artifacts. We fit a linear regression model on the years ranging from 2013 to 2023 and the percentage of papers with code and the percentage of code that compiled. From our analyses, we find an R^2 value of 0.88 (artifacts available) with a slope in a 95% confidence interval between 3.49 and 5.73, and an R^2 of 0.92 (artifacts that compiled) with a slope in a 95% confidence interval between 1.48 and 2.63. Thus, indicating that there is a significant trend in the growth of artifacts and compilable artifacts at applied security conferences.

Observations. Our analysis shows that when the provided artifacts are made for reproducibility, it increases the likelihood of reproducing the results. We find that of the artifacts made for reproducibility 45% compiled with minimal edits. For example, Alder et al.'s Faulty Point Unit [10] contains scripts to reproduce each table and figure. The instructions to set up and run are concise, and each script is labeled with the exact claim it reproduces. Visualizations within the instructions aided us when compiling de Carné de Carnavalet et al. [13].

Limitations exist within a portion of the artifacts to compile or directly reproduce results. For some, the incorrect metric is outputted (e.g., F1 score is reported in the paper, but the artifact outputs accuracy). Other challenges exist within packaging and dependencies. We find that some repositories require package versions that do not exist or are deprecated. Although most repositories do not adversely alter the machine they are run on, we did find a repository that turns off address space layout randomization (ASLR) and does not turn it back on. Thus, the artifact introduces a severe security vulnerability on the computer that reproduces the experiments. Further, this was not disclosed in the ReadMe.[4] A lack of instructions can hinder reproducibility, as we are able to compile and run 78 of 167 (48%) repositories with complete instructions, compared to only 20% of artifacts that had an empty ReadMe.

While common errors include missing code pieces or data, we did find several additional problems when trying to compile and run the code. We find that over 65% of repositories contain unclear output. For example, one artifact only outputs a time to run (e.g., performance), but the paper does not claim anything about the time to run. Further, no extra files are created, and the instructions are unclear on how to produce results. Some code repositories require manual changes to numerous variables. In one example, a repository path changes throughout the whole code base with no clear directions on what to change it to. In another, we find that the script to reproduce the results contained numerous errors, but the tool itself compiled and ran against the test cases. Interestingly, we found several repositories with Docker images that do not build. When we try to manually run the code, we are unable to do so due to errors within the code. One repository contained code without instructions or a pre-processing script. Other problems came from relying on other code. For example, one project relies on code from another developer that changed the API since the publication of the artifacts. Often, we find that we go over the one-hour limit for debugging or the 10-hour limit for training. One repository states that the code could run on a CPU in 30 minutes, but we are unable to get results after 10 hours.

Takeaway. *A significant portion of research artifacts still suffer from compilation issues, unclear outputs, dependency problems, and a lack of clear instructions, hindering reproducibility.*

4 Effects of AECs

The purpose of AECs is to increase the state of reproducibility by evaluating a published paper's artifacts and ensuring they run and reproduce the claims made in the paper. With the hope that any artifact that undergoes the AEC evaluation will work long after

[4]We have informed the authors of this by email twice and received no response.

publication. In this section, we explore how AECs affected the state of artifacts over time and what factors contributed to the success or failure to meet their stated goals.

4.1 Artifact Evaluation Committees

We explore the differences between the two conferences that have an AEC, ACM WiSec and ACSAC. Both conferences introduced AECs in 2017, allowing for a direct comparison. First, we look at the participation in AECs. Second, we compare the reproducibility of the artifacts across the two conferences over time. Third, we look at the badging of the artifacts that went through the AEC process.

Participation. In Figure 5, we see whether for each year the artifact compiles and/or runs, grouped by whether the artifact went through the AEC process. Overall, 57.7% of artifacts at WiSec and 79% of artifacts at ACSAC went through the AEC process during the years 2017-2023. We compute a confidence interval to determine if a significant difference exists between the rate of submissions to the AECs at ACSAC and WiSec. We use the confidence interval to show the direction and magnitude of the effect instead of computing a test statistic. Using Equation 1 we calculate a confidence interval for $(\hat{p}_1 - \hat{p}_2)$ where $\hat{p}_1$ = ACSAC's AEC submission rate, $\hat{p}_2$ = WiSec's AEC submission rate, n_1 = All ACSAC papers, n_2 = All WiSec papers, and with a 95% confidence level $z = 1.96$.

$$CI = (\hat{p}_1 - \hat{p}_2) \pm z(se), \quad se = \sqrt{\frac{\hat{p}_1(1-\hat{p}_1)}{n_1} + \frac{\hat{p}_2(1-\hat{p}_2)}{n_2}} \tag{1}$$

The resulting confidence interval for $(\hat{p}_1 - \hat{p}_2) = [0.17, 0.42]$. Because the confidence interval is entirely above zero, it provides statistical evidence that the rate of submission to ACSAC AECs is greater than the rate of submission to WiSec AECs. The difference is at least 17% but could be as much as 42%. Thus, there is a difference in the participation between the two conferences.

Working Code. Although increasing participation at the AECs is indicative of an improvement, whether AECs have resulted in more reproducible code remains an open question. Using Equation 1, we will compare the compilation and/or run rates between those submitted to an AEC and those not submitted to an AEC for each conference. For both conferences, we use $\hat{p}_1$ as submitted to the AEC and $\hat{p}_2$ as not submitted to the AEC. We get the 95% confidence interval of $[0.07, 0.54]$ for ACSAC and $[-0.18, 0.27]$ for WiSec. Thus, there is statistical evidence that an artifact is more likely to compile and run if it went through ACSAC's AEC. WiSec's confidence interval does not indicate that there is a difference. Thus, ACSAC's AEC is affecting the quality of a research artifact.

In prior work [20], Olszewski et al. conducted a reproducibility study of ML research papers on the AEC of USENIX Security that first appeared in 2020. In Figure 5, we provide their numbers on AEC participation and whether the code runs or not. Similar to trends in ACSAC, Olszewski et al. observed a growing trend in the AEC participation and evidence of an increase in runnable code. However, they were unable to conduct any meaningful statistical evaluation due to a lack of data. A further comparison and discussion around ML reproducibility is conducted in Section 5.1.

Takeaway. *Introducing AECs does not inherently improve the state of artifacts at a given conference.*

Badging. AECs award varying badges to artifacts that are evaluated, Artifacts Available, Artifacts Functional, and Results Reproduced. A paper may have more than one badge, and we discuss the trends of individual badges. There are only 20 artifacts with the Available badge, 140 with Artifact Functional, and 22 with Artifact Reproduced. As seen in Figure 6, of those artifacts available, we find that 20% of the artifacts compiled (10% with no edits and 10% with minimal edits). 26% required extra hardware or licenses, 11% reached our maximum time limit, and we are unable to compile 43%. Of the 140 functional artifacts, 52% compiled and ran, 36% did not compile, 21% required extra hardware, and 1% reached the maximum time limit. We noticed several discrepancies between the badges and our results. In one example, the paper has all three badges of Available, Functional, and Reproduced, yet the artifacts found online are only supplementary figures (i.e., the figures produced were not for the major claims of the paper). In other cases, the artifact is labeled as Available, but an empty GitHub repository is linked. In the case when an artifact had a higher level of reproducibility, we found the last commit to the repository was after the submission date for the AEC.

Of the 22 Reproduced badges within ACSAC and WiSec, we can compile and run 48% of the repositories without any edits, 19% compiled after minor edits, 10% required hardware or software, and 24% did not run. The two primary reasons we were unable to compile "Artifacts Reproduced" artifacts are due to unreleased datasets required to run the code or extra hardware (e.g., Apple Airtag, Android devices, or software-defined radios). We found evidence of data not being released for privacy concerns (e.g., user data) and evidence of missing data files within the artifact.

Although we do not always attain the level of reproducibility the AEC found, in some cases, we exceed it. For example, Alder et al. [10] only received the Functional badge, but we are able to reproduce all of their claims. Sebastián et al. [25] similarly only received the Functional badge, but the artifacts output the same precision and recall stated in the paper. Most likely, this discrepancy is due to the level of badge requested to be evaluated. For example, when submitting to an AEC, sometimes the AEC will ask for which levels to evaluate at. These papers could have asked for Functional instead of Reproduced. A possible avenue for future growth for conferences is retroactively updating the badges awarded to papers as improvements are made.

Takeaway. *The badges awarded to a paper do not always reflect the state of the artifact due to inconsistencies in evaluation processes.*

4.2 Differences by Conference

In Section 4.1, we find that there is statistical evidence of a difference in AEC participation between ACSAC and ACM WiSec. We explore some of the factors that could lead to the differences. We discuss the announcement of the call for artifacts (CFA) and look at the artifact evaluation process.

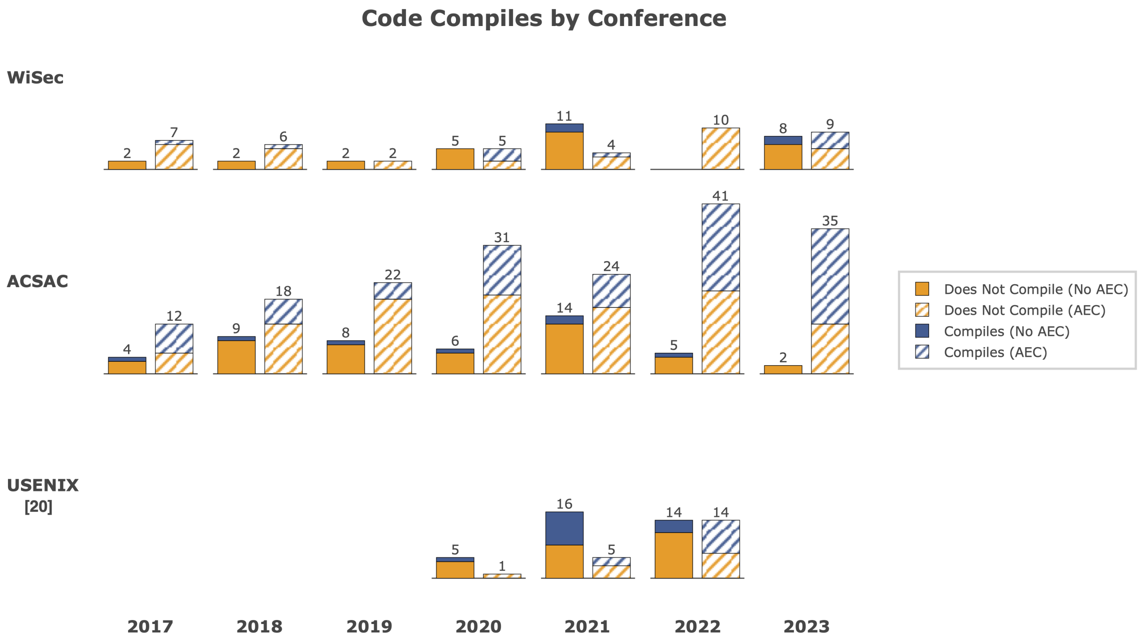

Figure 5: This figure shows the number of papers submitted to the AEC, not submitted to the AEC, and whether they compiled or not. For comparison, we include the numbers from Olszewski et al. [20] from USENIX Security.

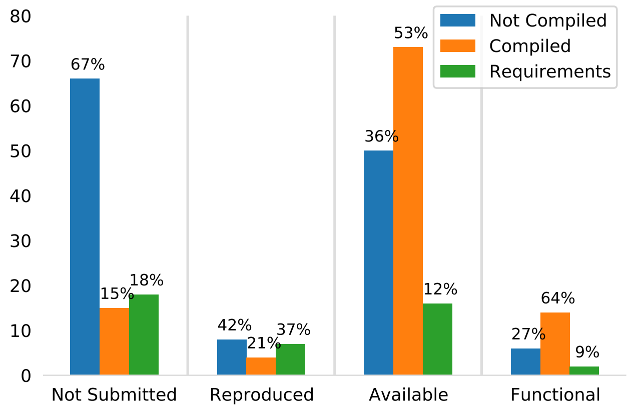

Figure 6: Of the papers in WiSec and ACSAC from 2017-2023, we show whether the artifact compiles, does not compile, or cannot run due to constraints in hardware by the AEC badge

The placement of the CFA varied between the two conferences. In 2017 through 2019, WiSec's CFA was a tab on the website labeled "Replicable Label" near the bottom of the menu sidebar. In 2020, the CFA is included as a tab directly beneath the call-for-papers (CFP) with a subsection in the CFP dedicated to the artifact evaluation submission. ACSAC in 2017 to 2018 included the CFA within the CFP under the paper submission guidelines. In 2019, the CFA was placed on a tab under the CFP.

The biggest difference between the ACSAC and WiSec AECs is the process of the artifact evaluation. The WiSec evaluation process has authors submit their artifacts. The AEC will then attempt to reproduce their results and notify the authors of the AEC's results, then all artifacts that went through the process are placed on the website. The instructions on WiSec's website describe how to prepare the artifact submission by including: instructions to prepare a VirtualBox VM, a script to generate each object in the paper, a ReadME, and a link to download the VM. In contrast, ACSAC's artifact evaluation is a month-long, iterative process between the artifact evaluators and the authors. The AEC will attempt to follow the instructions provided by the authors and notify the authors of any difficulties or discrepancies. The authors then have multiple opportunities to iterate on their submitted artifact for the committee. We note that while this is recommended, it is not necessarily required of the authors. Included on ACSAC's CFA are instructions to submit an artifact. The guidelines include creating a single repository, a mapping of information in the artifact to the paper sections, containerizing the artifact, data inclusion, and time requirements to run each part of the artifact. Further, ACSAC requires authors to "commit to [submitted artifacts] permanently available online on a publicly accessible website" [4].

We note that this exploration is not exhaustive, but it identifies several factors that could have affected the performance of the AEC. There are numerous other confounding factors that result in a complex dynamic between AECs and PCs. One explanation for the

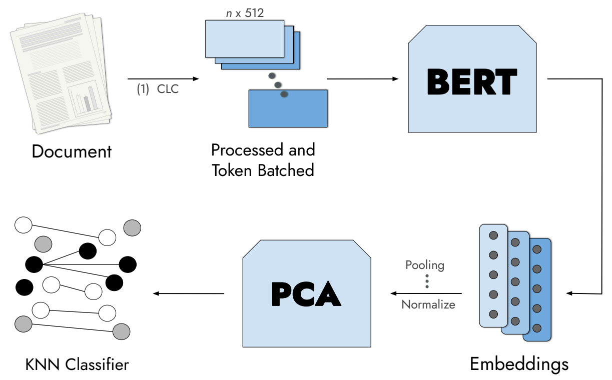

Figure 7: Our Clustering Pipeline: First, we pre-process and tokenize each document. Second, we embed the processed data using BERT. Third, we apply principal component analysis. Fourth, we cluster the documents using the top three principal components.

difference between ACSAC and Wisec we observed is that ACSAC grew in the number of papers, as WiSec maintains around the same accepted papers. This can result in both funding differences and overall staffing dedicated to evaluating artifacts. Further, we cannot account for factors such as email notifications that could have been sent to authors.

Takeaway. *The AEC process is just as important as implementing an AEC. Clear instructions to authors and an iterative processes can result in better outcomes from an AEC.*

5 Facets of Reproducibility

There are numerous ways to understand the reproducibility of artifacts. In this section, we characterize several facets of reproducibility, including: topic area, programming language, and funding agency. To do this, we first cluster each paper in our study to determine the sub-domain. Next, we analyze the reproducibility in the topic area and compare our results to prior work. Then, we compare by programming language and funding agency.

5.1 Topic Area

Clustering and Labeling. Some conference papers (e.g., AsiaCCS, ACSAC, and WiSec) have keyword and topic areas listed within each document.[5] We use these labels to classify the rest of the documents to discuss reproducibility by topic area. To assign topic areas for each paper, we use natural language processing to embed each document, reduce the dimensionality of the embeddings using principal component analysis (PCA), cluster the embeddings using hierarchical clustering, and assign labels to each cluster based on the majority vote of clusters. Our labeling methodology is shown in Figure 7. We download each conference paper as a PDF document. We use PyPDF2 [1] to extract the text from each document. We then clean each text corpus by removing stopwords, removing special characters, and adjusting line breaks. We extract each of the keywords from the papers that have keywords and store them as labels. Of the 2,009 papers in this study, 876 contain keywords, CCS concepts, or index terms. We remove "engineering" and "security" from the keywords labels.

[5]Not every paper at these conferences has keywords.

We use the BERT model [14] to embed each document. After each document is cleaned, we prepare each text corpus for embedding by lemmatizing the full document and batching the text corpus into $\lceil \frac{T}{512} \rceil$ input batches, where T is the total number of tokens (i.e., words) in each document and 512 is BERT's input shape. The final batch for each document is padded with a special [PAD] token. Thus, each document is broken into a total batch of $\lceil \frac{T}{512} \rceil \times 512$. The BERT model has 768 hidden states. Thus, we have an embedding of size $\lceil \frac{T}{512} \rceil \times 512 \times 768$. We then mean pool and normalize the document embedding to move the $\lceil \frac{T}{512} \rceil \times 512 \times 768$ to 1×768, as it is the best performance for the text clustering task [26]. Thus, after embedding and normalizing, we have a matrix of size $2,009 \times 768$ that represents the embedding of each document.

We reduce the dimensionality further by using principal component analysis (PCA). PCA is a change of basis such that the n-th principal component points in the n-th largest direction of variation. This is done by performing singular value decomposition on the covariance matrix of the data matrix. The n-th singular value corresponds to the amount of variance the n-th principal component describes. After performing PCA on the embedding matrix, we find that the first three principal components account for 81% of the variance (43%, 29%, and 9%) in the embedding matrix. Thus, we use the first three principal components for clustering, resulting in a $2,009 \times 3$ matrix.

We use spectral clustering, as it has great performance on BERT embeddings [21], to determine the topic areas of documents without keywords. Each document in the embedding space is its cluster. From there, we iteratively merge each cluster by considering the closest cluster by the minimum Euclidean distance between each cluster (e.g., in the case of single point clusters, the shortest distance between documents is merged). Taking the union of keywords for a cluster will indicate the domain area of the cluster. We then look at the top 10 clusters and select the topic of each by looking at the top 10 most frequent keywords within the union. After clustering, we end up with machine learning, cryptography, network security, hardware security, mobile devices, and systems security. For discussion, we treat these domains as disjoint sets, though a paper can have multiple domain areas.

Topic Area. We want to explore how reproducibility differs by sub-domain of computer security and privacy. After clustering and keyword extraction, we have five main clusters, excluding machine learning. We look at cryptography, network security, systems security, hardware security, and mobile security. The cluster topics are not disjoint, but we treat them as such to simplify the discussion.

After aggregating the groups, there are 932 papers within the five clusters that we look at. Each cluster consists of 318 for network security, 199 for systems security, 165 for mobile security, 125 for hardware security, and 125 for cryptography. We see in Figure 8, the total number of papers, the number of artifacts, and the number we are able to compile for each topic. Of the topics, we are able to

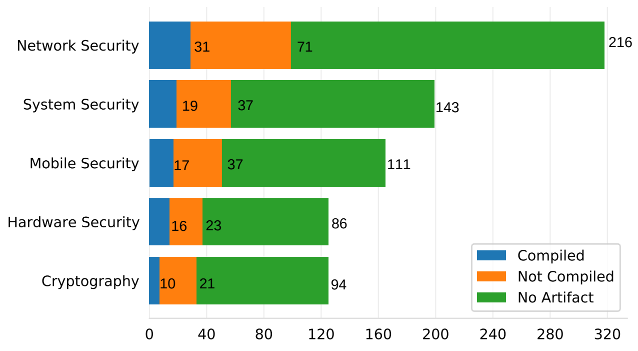

Figure 8: This shows the number of papers in each cluster, the number of artifacts, and the number of compilable artifacts.

compile 38% of the hardware artifacts, 33% of mobile security, 33% of system security, 29% of network security, and 21% of cryptography.

Machine Learning. Much of the work in machine learning outputs summary metrics as a gauge of performance. This inherently gives machine learning an advantage for computational reproducibility, as the outcomes of the research are direct results of the code running. For example, a paper claims that their machine learning model achieves 99% accuracy. The code artifacts to calculate that are a part of the code to use train and run the model.[6] We explore how reproducible machine learning papers are at applied security conferences, and whether there is a difference between prior work [20] and the applied security conferences, as Olszewski et al. [20] only collected data for machine learning.

Of the 2,009 papers that are in our study, 324 are machine learning papers, 125 from ACSAC, 115 from AsiaCCS, 55 from EuroS&P, and 26 from WiSec. 115 of the 324 (36%) papers provide artifacts. This is 4% less than what Olszewski et al. observed at other security conferences. 39% of the artifacts ran with minimal to no edits compared to 44% that ran in prior work [20]. We cannot run 5%due to hardware constraints. The conference with the highest rate of running machine learning artifacts is ACSAC at 49%. Although there are fewer artifacts within the machine learning papers at applied security conferences, there is a 6% increase in the running of these artifacts from ACSAC over other conferences.

Takeaway. *Reproducibility rates vary significantly across different sub-domains within computer security and privacy, with machine learning exhibiting comparatively higher rates due to its inherent output structure.*

5.2 Programming Language

There are over 20 different languages present in the artifacts. The most common languages in the artifacts are Python at 44%, C at 16%, and C++ at 11%. Other languages present within the dataset include Java, Javascript, Matlab, Rust, Go, and R. We find that only

[6]Consider a non-machine learning paper that creates a new protocol and claims a performance boost. Calculating the performance boost is not inherent to running the protocol. A separate script is required.

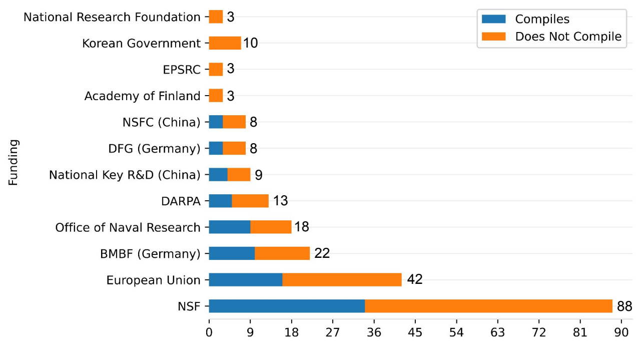

Figure 9: Reproducibility by funding agency.

54% of Python artifacts run, 70% of C artifacts compile and run, and 52% of C++ artifacts compile and run. Python is a growing language known for its simple syntax and data analysis packages. Thus, it is not surprising that Python is the most common language in the artifacts. Python offers several ways to provide packaging and dependencies within the code (e.g., `requirements.txt` or Anaconda environments), yet the most common problem stopping us from running Python repositories is the lack of explicit requirements. While there are benefits to the dependencies, versioning can become difficult if not managed by an environment. In one example, several versions of Python are required for the repository. Python 2.7, deprecated on January 1, 2020 [3], was needed for a package version, but Python 3 code styles are used throughout the repository. Python simplifies development processes, but this can also lead to very complicated environments and dependencies.

C and C++ account for approximately 27% of the artifacts. We find that the C artifacts compile at the highest rate amongst the top three languages. The C artifacts are often contained within a Docker image that we are able to build. While C and C++ can offer improvements in performance, code repositories and file structures can become complex. In one particular example, the code results in an infinite loop. Upon further inspection, a try/catch statement caused the artifact to continuously request a URL that is broken. In another, a ReadMe last modified eight years ago states, "[Our code] is not organized... will try to restructure ASAP."

Takeaway. *While Python is the most prevalent language in research artifacts, it also exhibits the lowest run success rate, largely due to dependency management issues, while C and C++, despite their inherent complexity, demonstrate higher compilation rates.*

5.3 Funding Agency

We collected the funding agency by identifying the grant that funded the publication directly from the Acknowledgements section of each paper to determine if there is a difference in reproducibility due to the funding agency. Funding agencies have an explicit interest in funded research resulting in reproducible artifacts, as this validates the research outcomes and maximizes the impact of their investment. When we consider the number of compilable and runnable artifacts by funding agency, we see in Figure 9 that the National Science Foundation (NSF) overwhelmingly funds the most artifacts, with 78 total and 23 that were successfully compiled (29%). Thus, while NSF-funded research produces a large volume of artifacts, the rate of successful compilation is not proportional to the number of artifacts produced. For example, the German Federal Ministry of Education and Research (BMBF) had 45% of 20 artifacts that compiled and ran. Following the NSF in terms of funding volume are the European Union (EU), BMBF, the Office of Naval Research (ONR), and the Defense Advanced Research Projects Agency (DARPA). The dominance of the NSF in both total and compiled artifacts is most likely due to the overall higher volume of funding they provide to research in this domain, but the lower compilation rate suggests further investigation into the consistency of artifact quality within NSF-funded projects.

6 Discussion

6.1 Stakeholders

Despite some steps taken towards improvement, the evidence presented in this study points towards a continued challenge in reproducibility in the applied security community. Our work is not a criticism of any work or author; it is a reflection on the state of reproducibility in the security community and the identification of tangible improvements to be made. Moving towards reproducible research does not solely require a shift in how research is conducted. We encourage not only changes in the methodological process but also in shifts from funding agencies and their outcomes for research, program committees and how papers are evaluated, and authors in building transparent research. We highlight changes that could be made among researchers, conferences, and funding agencies.

Researchers: Reproducible research is not a passive state, but an active effort from researchers. We recognize the difficulty of building reproducible artifacts and applaud all researchers who have made efforts to publish their artifacts. The benefits of reproducibility should encourage all researchers to make experimental artifacts available. Reproducibility provides tangible benefits to research teams. Designing artifacts with reproducibility as a core principle allows for easy transitions between personnel on a project (e.g., a student graduating). When building artifacts for your research, anticipate issues that can arise when trying to set up your code and understand the limitations that exist within each setup. For example, requiring expensive hardware to run the artifact will make it difficult to reproduce. Further, consider how the artifacts will run several years in the future. A common issue we found was that several artifacts required specific versions of packages that no longer existed, and there was no option to build from source.

Conferences: We see how minor shifts in the publication process (e.g., the introduction of AECs) result in a change in reproducibility. For the past two years, over 90% of available artifacts at ACSAC went through the artifact evaluation process, and as discussed in Section 3, there is a significant growth in the number of artifacts and artifacts that compile and run. This phenomenon was not present for WiSec. Although WiSec saw a growth in participation, the artifacts were not more likely to run if they went through the AEC. This is most likely due to the difference in how the AECs operate. ACSAC's artifact evaluation process is interactive and iterative, whereas WiSec's artifact evaluation process is static with no editing of the original artifact. Thus, it is imperative to create guiding principles for the AECs and identify the outcomes and benefits not just for the AEC but also for the program committee as well. Conferences must then communicate the results of the AEC and be open and transparent about the outcomes.

It has been noted previously that this increases the demand for resources to successfully run an AEC [11]. Further improvement could move the AEC to take place during the review process. For example, after a paper passes round one of the review process, it might then go to the AEC to validate the research before making it to round two. Although this may not be scalable in modern conference design. As discussed in Section 4, we found papers with higher levels of reproducibility than the awarded badges. We recognize that these changes are not without difficulty, but structured guidelines and reflective analysis can result in drastic improvement to the state of artifacts.

Funding Agencies: Ensuring the reproducibility of funded projects increases the benefits of the research and offers tangible outcomes for funding agencies. Thus, reproducibility increases the credibility of research outputs. From the perspective of funding agencies, the promotion of reproducibility aligns with their overarching goals of supporting high-quality research. As discussed in Section 5, the National Science Foundation supports the most artifacts that we were able to compile and run, but this was not the largest proportion of artifacts. Reproducibility aids in transitioning academic research to industrial applications. When research findings are reproducible, they are more readily translatable into practical solutions and technologies, facilitating a transition from laboratory settings to real-world implementations. We encourage funding agencies to adopt reproducibility within their core principles.

Reproducibility is not a burden solely on the authors; it requires changes throughout the research cycle, including researchers, conferences, and funding agencies. We encourage the reader to view reproducibility not as a challenge, but as an opportunity for the security community to improve transparent and credible research.

6.2 Challenges

Prior work [20, 24] focus on reproducibility in a sub-domain of computer security and privacy. Adapting a framework that extends both the results of the study and provides sufficient definitions remains an open problem. For example, quantifying data with train/test splits is explicit in machine learning, but explicitly defining measures of data across several domains is more difficult. Further, evaluating methodological procedures in an indirect study of machine learning papers can be objective by looking at the various algorithms and identifying whether each hyper-parameter is defined. When abstracting to all security domains, it requires domain expert knowledge of the methodology to be able to perform an indirect study. Thus, aggregating information may lose meaning across the topic areas of computer security and privacy.

While prior work [15] attempts to provide a thought process for abstracting the experimental process of machine learning papers, there remains a lack of practical guidance to apply a broader framework to reproducibility studies. We focus on the objective outcomes we can still study without losing meaning. For example, we score a ReadMe on having instructions to run the code and a description of each file in the repository. This provides objective bounds for the experiments we run.

6.3 Limitations

We identify the limitations of this work and how we addressed biases that could exist within our work. Our work, similar to others [12, 20, 22], measure reproducibility as a binary scale; we often only report results for whether or not the code runs. We give each repository the best case (e.g., if the repository only recreates a claim of the paper, we demarcate it as reproduced). Thus, it could overstate the performance of a repository. We limit the amount of time we spend on each artifact similar to previous work [12, 20, 22]. This may not accurately reflect the full potential of a repository, but we often found that the 10-hour limit for downloading data or running the artifact was often not the limiting factor.

The most difficult challenge in this study was the significant effort required to locate, understand, and attempt to execute the diverse range of submitted artifacts, often with incomplete or outdated documentation. Furthermore, the frequent lack of standardized environments and the reliance on specific, sometimes unavailable, hardware or software configurations severely hampered our ability to consistently evaluate the reproducibility claims. Future testbeds may help to alleviate this problem. For example, the SPHERE [2] project is building scalable research hardware that can be configured by security researchers to suit their needs. This will not only provide valuable resources to the community but also mitigate an inhibitor to reproducibility.

7 Conclusion

The call for reproducible research within the computer security and privacy community results in ACSAC and ACM WiSec introducing the first AECs in the security community. However, no prior study looks at the reproducibility of experimental artifacts published at these venues. As such, we conduct a large-scale reproducibility study collecting artifacts from over 11 years of applied security conferences. We identify that although AECs have been introduced, only ACSAC's AEC has resulted in statistical evidence for improved AEC participation and artifact quality. Thus, indicating that improved reproducibility is achievable by implementing interactive artifact evaluation processes.

8 Acknowledgements

This work was supported in part by the National Science Foundation grants CNS-2446321 and CNS-2206950. Any findings and opinions expressed in this material are those of the authors and do not necessarily reflect the views of the funding agencies. Finally, the authors would like to thank the members of the Artifact Evaluation Committee for their efforts in improving our open-source artifact.

References

[1] [n. d.]. PyPDF2. https://pypi.org/project/PyPDF2/.
[2] [n. d.]. Security and Privacy Heterogeneous Environment for Reproducible Experimentation. https://sphere-project.net/.
[3] [n. d.]. Sunsetting Python 2. https://www.python.org/doc/sunset-python-2/.
[4] 2017. ACSAC Call for Artifacts. https://www.acsac.org/2017/artifacts/.
[5] 2017. WiSec Artifactiation Evaluation Committee 2017. https://wisec2017.ccs.neu.edu/reproducibility.html.
[6] 2020. Artifact review and badging - current. https://www.acm.org/publications/policies/artifact-review-and-badging-current
[7] 2020. USENIX Security '20 Call for Artifacts. https://www.usenix.org/conference/usenixsecurity20/call-for-artifacts.
[8] 2021. USENIX Security '21 Call for Artifacts. https://www.usenix.org/conference/usenixsecurity21/call-for-artifacts.
[9] 2023. SIGSAC ACM CCS Call For Artifacts. https://www.sigsac.org/ccs/CCS2023/call-for-artifacts.html.
[10] Fritz Alder, Jo Van Bulck, David Oswald, and Frank Piessens. 2020. Faulty point unit: ABI poisoning attacks on Intel SGX. In *Proceedings of the 36th Annual Computer Security Applications Conference*. 415–427.
[11] Terry Benzel. 2023. Security and Privacy Research Artifacts: Are We Making Progress? *IEEE Security & Privacy* 21, 01 (2023), 4–6.
[12] Christian Collberg, Todd Proebsting, and Alex M Warren. 2015. Repeatability and benefaction in computer systems research. *University of Arizona TR* 14, 4 (2015).
[13] Xavier de Carné de Carnavalet and Mohammad Mannan. 2014. Challenges and implications of verifiable builds for security-critical open-source software. In *Proceedings of the 30th Annual Computer Security Applications Conference*. 16–25.
[14] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).
[15] Odd Erik Gundersen. 2021. The fundamental principles of reproducibility. *Philosophical Transactions of the Royal Society A* 379, 2197 (2021), 20200210.
[16] Imran Hossen and Xiali Hei. 2022. aaecaptcha: The design and implementation of audio adversarial captcha. In *2022 IEEE 7th European Symposium on Security and Privacy (EuroS&P)*. IEEE, 430–447.
[17] Simon Malcomber, Margaret Martonosi, James L. Moore, Susan S. Margulies, Alexandra R. Isern, Alicia J. Knoedler, Kendra V. Sharp, Sean L. Jones, Kellina M. Craig-Henderson, and Erwin Gianchandani. 2022. Dear Colleague Letter: Reproducibility and Replicability in Science. (22 Oct 2022).
[18] National Academies of Sciences Engineering and Medicine and others. 2019. *Reproducibility and replicability in science*. National Academies Press.
[19] Daniel Olszewski, Allison Lu, Carson Stillman, Kevin Warren, Cole Kitroser, Alejandro Pascual, Divyajyoti Ukirde, Kevin Butler, and Patrick Traynor. [n. d.]. Artifacts for "Get in Researchers; We're Measuring Reproducibility". https://github.com/reproducibility-sec/reproducibility.
[20] Daniel Olszewski, Allison Lu, Carson Stillman, Kevin Warren, Cole Kitroser, Alejandro Pascual, Divyajyoti Ukirde, Kevin Butler, and Patrick Traynor. 2023. "Get in Researchers; We're Measuring Reproducibility": A Reproducibility Study of Machine Learning Papers in Tier 1 Security Conferences. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*. 3433–3459.
[21] Alina Petukhova, Joao P Matos-Carvalho, and Nuno Fachada. 2024. Text clustering with LLM embeddings. *arXiv preprint arXiv:2403.15112* (2024).
[22] Edward Raff. 2019. A step toward quantifying independently reproducible machine learning research. *Advances in Neural Information Processing Systems* 32 (2019).
[23] Michael Rodler, David Paaßen, Wenting Li, Lukas Bernhard, Thorsten Holz, Ghassan Karame, and Lucas Davi. 2023. EF CF: High Performance Smart Contract Fuzzing for Exploit Generation. In *2023 IEEE 8th European Symposium on Security and Privacy (EuroS&P)*. IEEE, 449–471.
[24] Moritz Schloegel, Nils Bars, Nico Schiller, Lukas Bernhard, Tobias Scharnowski, Addison Crump, Arash Ale Ebrahim, Nicolai Bissantz, Marius Muench, and Thorsten Holz. 2024. SoK: Prudent Evaluation Practices for Fuzzing. *arXiv preprint arXiv:2405.10220* (2024).
[25] Silvia Sebastián and Juan Caballero. 2020. Avclass2: Massive malware tag extraction from av labels. In *Proceedings of the 36th Annual Computer Security Applications Conference*. 42–53.
[26] Alvin Subakti, Hendri Murfi, and Nora Hariadi. 2022. The performance of BERT as data representation of text clustering. *Journal of big Data* 9, 1 (2022), 15.