

USENIX

THE ADVANCED COMPUTING SYSTEMS ASSOCIATION

Not All Those Who Share Are Lost: Analyzing 25 Years of Cybersecurity Artifact Sharing Practices Through Automated Discovery

Daan Vansteenhuyse, Arthur Bols, and Lieven Desmet, *DistriNet, KU Leuven;* Victor Le Pochat, *Independent;* Jo Van Bulck and Marton Bognar, *DistriNet, KU Leuven*

https://www.usenix.org/conference/usenixsecurity26/presentation/vansteenhuyse

This artifact appendix is included in the Artifact Appendices to the Proceedings of the 35th USENIX Security Symposium and appends to the paper of the same name that appears in the Proceedings of the 35th USENIX Security Symposium.

August 12–14, 2026 • Baltimore, MD, USA

ISBN 978-1-939133-58-8

Open access to the Artifact Appendices to the Proceedings of the 35th USENIX Security Symposium is sponsored by



جامعة الملك عبدالله للعلوم والتقنية

King Abdullah University of Science and Technology



USENIX Security '26 Artifact Appendix: Not All Those Who Share Are Lost: Analyzing 25 Years of Cybersecurity Artifact Sharing Practices Through Automated Discovery

Daan Vansteenhuyse[1], Arthur Bols[1], Lieven Desmet[1], Victor Le Pochat[2], Jo Van Bulck[1], Marton Bognar[1]

[1]DistriNet, KU Leuven
[2]Independent

A Artifact Appendix

A.1 Abstract

This artifact consists of three parts: (1) the ArtiFinder tool, capable of identifying artifact URLs in paper PDFs; (2) the output of ArtiFinder on 25 years of top-tier cybersecurity papers together with metadata such as authors and citation counts; and (3) scripts to reproduce our analysis and generate the figures from our paper. All scripts can be executed on a commodity UNIX-based machine with internet access.

A.2 Description & Requirements

A.2.1 Security, privacy, and ethical concerns

ArtiFinder identifies artifact URLs in published paper PDFs. Due to licensing constraints, our artifact does not contain the PDFs used to generate the full dataset. The metadata contains a link to all papers, allowing users to download the PDFs for subsequent experimentation with ArtiFinder. Our scripts run without modifying local data.

A.2.2 How to access

The initial version of ArtiFinder and our collected dataset is archived at https://doi.org/10.5281/zenodo.20412201. This deposit also includes all scripts necessary to perform our analysis.

We are continuing the development of ArtiFinder at https://github.com/DistriNet/ArtiFinder, where we invite contributions. We also maintain an active version of the dataset at https://github.com/DistriNet/ArtiFinder-Data, open to corrections and extensions.

A.2.3 Hardware dependencies

None.

A.2.4 Software dependencies

Our artifact can be executed on any UNIX-based system with Python 3.12 and Poppler[1].

A.2.5 Benchmarks

The artifact does not include the analyzed PDF files, but our raw data files in `data/` contain all metadata used in the analysis, together with links to the papers. Our tool ArtiFinder is designed to work with any paper PDF.

A.3 Set-up

A.3.1 Installation

We recommend using a Debian-based system, but any UNIX-based system should work, provided it has the software installed mentioned in Appendix A.2.4. Download the files from Zenodo or pull our GitHub repository and (optionally) set up a virtual Python environment.

For example, on Debian 12, the installation can be performed using the following commands:

```
sudo apt-get install python3-gi \
python3-gi-cairo gir1.2-gtk-4.0 libxml2 cmake \
pkg-config libcairo2-dev girepository-2.0 \
libpoppler-glib-dev gir1.2-poppler-0.18
python3 -m venv venv
source venv/bin/activate
pip install -r requirements.txt
```

Some heuristics use the GitHub API to look up metadata, it is therefore recommended to create a personal access token and add it to `.env` as shown in `.env.example`. This token can be created for free at https://github.com/settings/personal-access-tokens.

[1] https://poppler.freedesktop.org/

A.3.2 Basic Test

`python test_install.py` checks if all the required packages are installed and should not return any output.

A.4 Evaluation workflow

A.4.1 Major Claims

(C1): ArtiFinder is capable of extracting links from a paper PDF and ranking them to identify the most likely artifact link. This is shown by experiment E1 and is described in Section 3.

(C2): ArtiFinder is able to identify artifacts links in cybersecurity paper PDFs with a high (93.3–98.8%) accuracy compared to three different ground truth datasets. By conducting an ablation study, we show that multiple heuristics are needed to achieve this accuracy. This is shown by experiment E2 and is described in Section 3.3.

(C3): By running ArtiFinder on a corpus of 9,054 papers, we identify trends and statistics in artifact sharing across top-tier cybersecurity literature. All statistics and figures are generated in experiment E3 and are described in Section 4.

A.4.2 Experiments

(E1): [5 human-minutes + 2 compute-minutes + 10MB disk]:
How to: Due to licensing constraints, we cannot release paper PDFs. However, ArtiFinder works on any paper provided in PDF format, which the user can provide. Our raw data (`data/data.json`) contains links to all the papers analysed in our research.
Preparation: Ensure all Python dependencies are installed and the paper PDF is downloaded. The necessary metadata can be inputted directly into the CLI or be formatted as a JSON file containing the fields: title, authors, year and conference.
Execution:
```
python artifinder-cli.py \
--pdf /path/to/paper.pdf \
--json /path/to/metadata.json \
-o /path/to/output.json
```
Optionally, the metadata can be added using `--authors`, `--year`, `--conf`, `--title`.
Results: The output is written to the supplied output file and contains the metadata extended with the extracted links and the discovered artifact(s).

(E2): [1 human-minute + 5 compute-minutes + 200MB disk]:
How to: By comparing the raw ArtiFinder output with groundtruth datasets, the accuracy numbers can be recalculated.
Preparation: Install all Python dependencies.
Execution:
```
python reproduce_results.py \
        --experiment E2
```
Results: The output can be found in `results.md`, which should match the numbers in the paper and `expected_results.md`.

(E3): [1 human-minute + 5 compute-minutes + 200MB disk]:
How to: Run the analysis scripts to reproduce the various claims made about our dataset.
Preparation: Install all Python dependencies.
Execution:
```
python reproduce_results.py \
        --experiment E3
```
Results: The output can be found in `results.md` which should match the numbers in the paper and `expected_results.md`. The output is formatted per paper (sub)section. Generated figures are stored in `/figs`.

A.5 Notes on Reusability

ArtiFinder is designed to be modular, allowing adaptations with more heuristics depending on the use-case. On our GitHub, we invite contributions to optimize our artifact discovery process, possibly by fine-tuning it for other computer science fields. We hope ArtiFinder allows researchers to better map artifact availability trends to promote open and reproducible science. At https://github.com/DistriNet/ArtiFinder-Data, we release our dataset, open for corrections of mistakes made by ArtiFinder.

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/usenixsec2026/.