PolyDoc: Surveying PDF Files from the PolySwarm Network

Prashant Anantharaman, Rob Lathrop, Bx Shapiro, Michael E. Locasto

LangSec 2023
25th May 2023

prashant.anantharaman@narfindustries.com

Malware in PDFs

https://eternal-todo.com/blog/cve-2013-2729-exploit-zeusp2p-gameover

How well-formed are these malicious PDF files used in large-scale phishing campaigns?

Summary

- We compare results of various threat engines with popular PDF tools
- We use a tracer to better understand and explain parse failures
- We evaluate *PolyDoc* on a large corpus of malicious PDF files

Well-formed vs. Malformed

Considers the errors produced by the four PDF tools we study

Benign vs. Malicious

Considers the PolyScore values provided by the PolySwarm API

0.2-0.5: Benign
0.7-1.0: Malicious

Are malicious PDFs malformed?

PolySwarm network

- Crowdsources malware threat intelligence
  - Total of 49 engines listed online
- Some of these engines are targeted: sometimes trained to detect only Ransomware and phishing attempts, whereas others may look for malicious URLs
- The API returns a PolyScore (from 0.2 to 1), and a malware label

Summary of Findings

- 60% of the files we scanned from the PolySwarm network received PolyScores of 0.8 and over.

- PolySwarm also provides a malware class: we found that certain classes such as Cryptominer and Trojan manifest with specific syntax errors found by PDF tools.

- The PDF error ontology does not capture several errors that we encountered while running PolyDoc on PDFs from PolySwarm

Outline

- **Design Space Overview**
- PolyDoc Design
- Findings
  - Baseline experiments
  - PolySwarm experiments
- Current and Future Directions

Parser Tracing Frameworks

- Parsers contain their own virtual machines, and instrumenting just the parser logic is challenging
- Malware “detonating” mechanisms focus on the malware payload more often than the delivery mechanism (buffer overflows and syntactic malforms)
- Control-flow analysis approaches, such as PolyTracker, may produce *byte accountings*

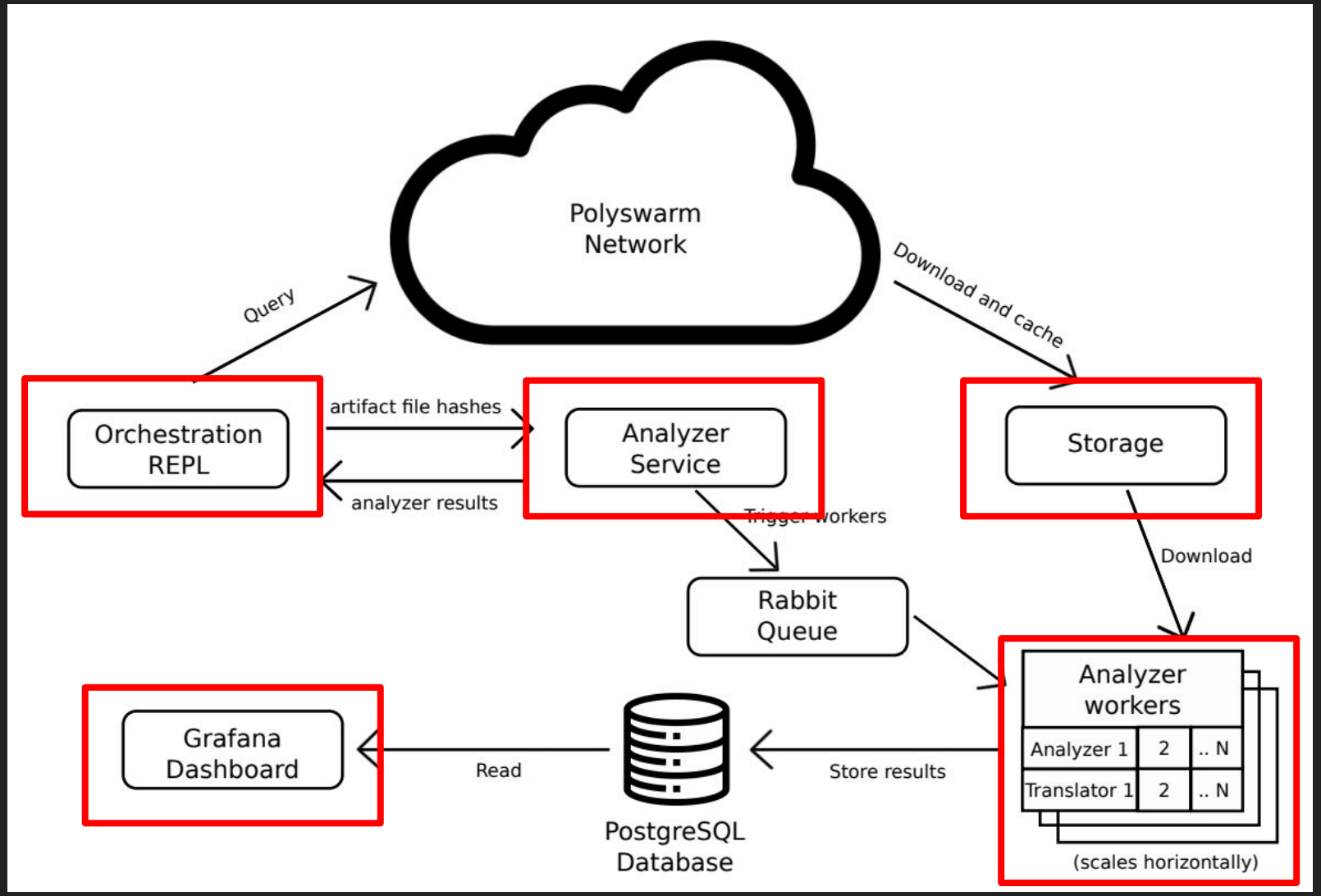
bpftrace

- kernel, syscall, library, and application-level tracing
- Provides a near-optimal point in the design space of tracing fidelity vs. performance
  - Executes in kernel space,
  - No simulation/emulation costs,
  - and no unneeded traps
- Supports the use of a scripting framework

Outline

- Design Space Overview
- **PolyDoc Design**
- Findings
  - Baseline experiments
  - PolySwarm experiments
- Current and Future Directions

System Architecture



Selected PDF tools

Tool	Version	Command
Caradoc[a]	0.3	`caradoc extract --verbose --decode-streams --relax-streams {file}`
Mutool[b]	1.18.0	`mutool clean -s -d -i -f {file}`
Poppler[c]	0.84.0	`pdfinfo -box -meta -js -struct -struct-text -isodates -dests {file}`
Pdftools[d]	0.7.4	`pdf-parser.py -v -O {file}`

- PDF Tools (Didier Stevens) displays data in PDF objects and other metadata in the PDF file
- Caradoc (LangSec ‘16) is a strict implementation of the PDF specification enforcing various syntactic, type, and graph constraints

Outline

- Design Space Overview
- PolyDoc Design
- **Findings**
  - **Baseline experiments**
  - **PolySwarm experiments**
- Current and Future Directions

Evaluation

- What PolyScores do we find on GovDocs files that are known to be clean and well-formed?
- What malformation categories from the PDF Error Ontology do we see in malicious files?

Baseline Experiment

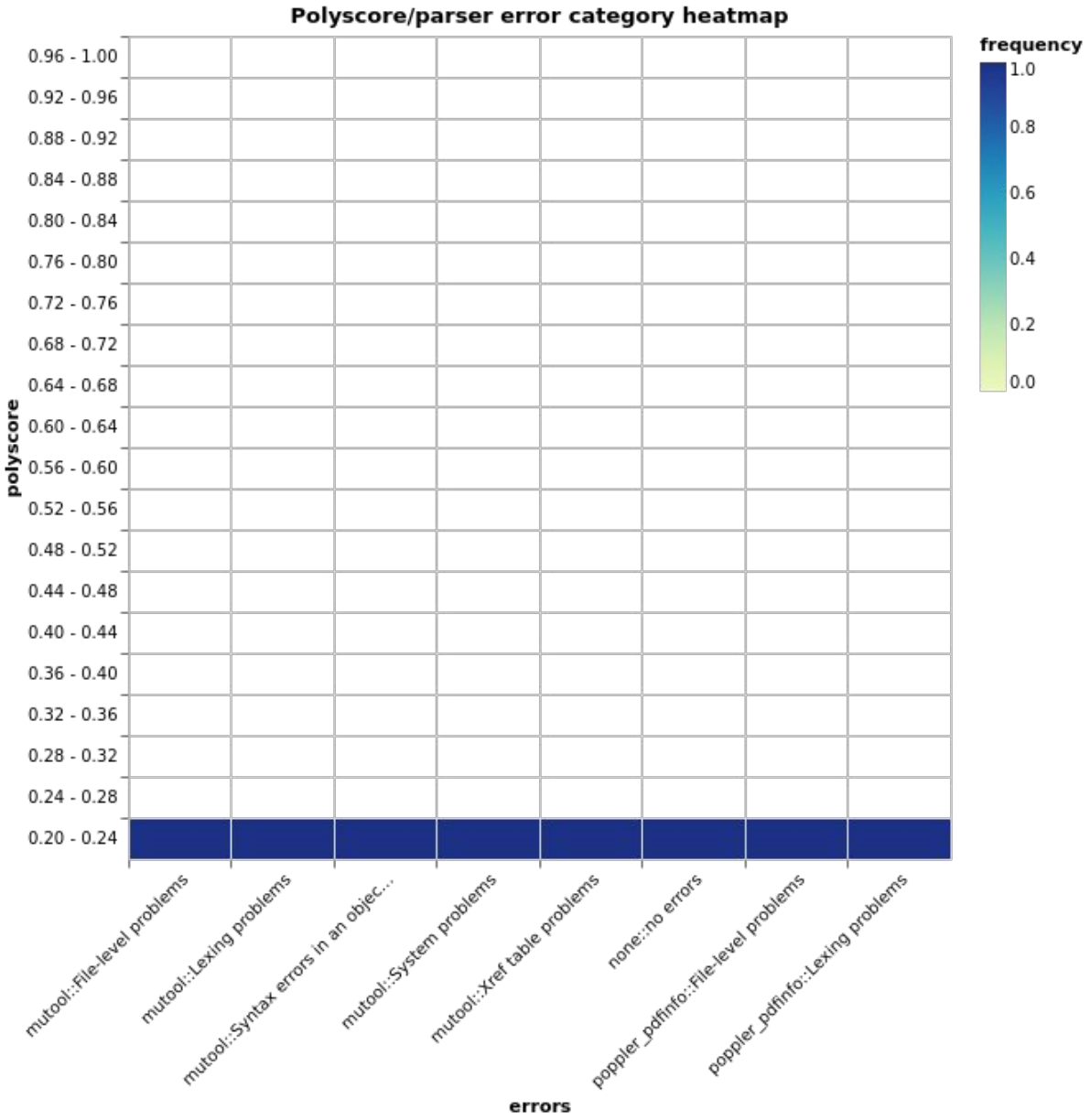
- Use the GovDocs Dataset
- Select known *clean* files
- Run these files through PolySwarm and the set of selected parsers

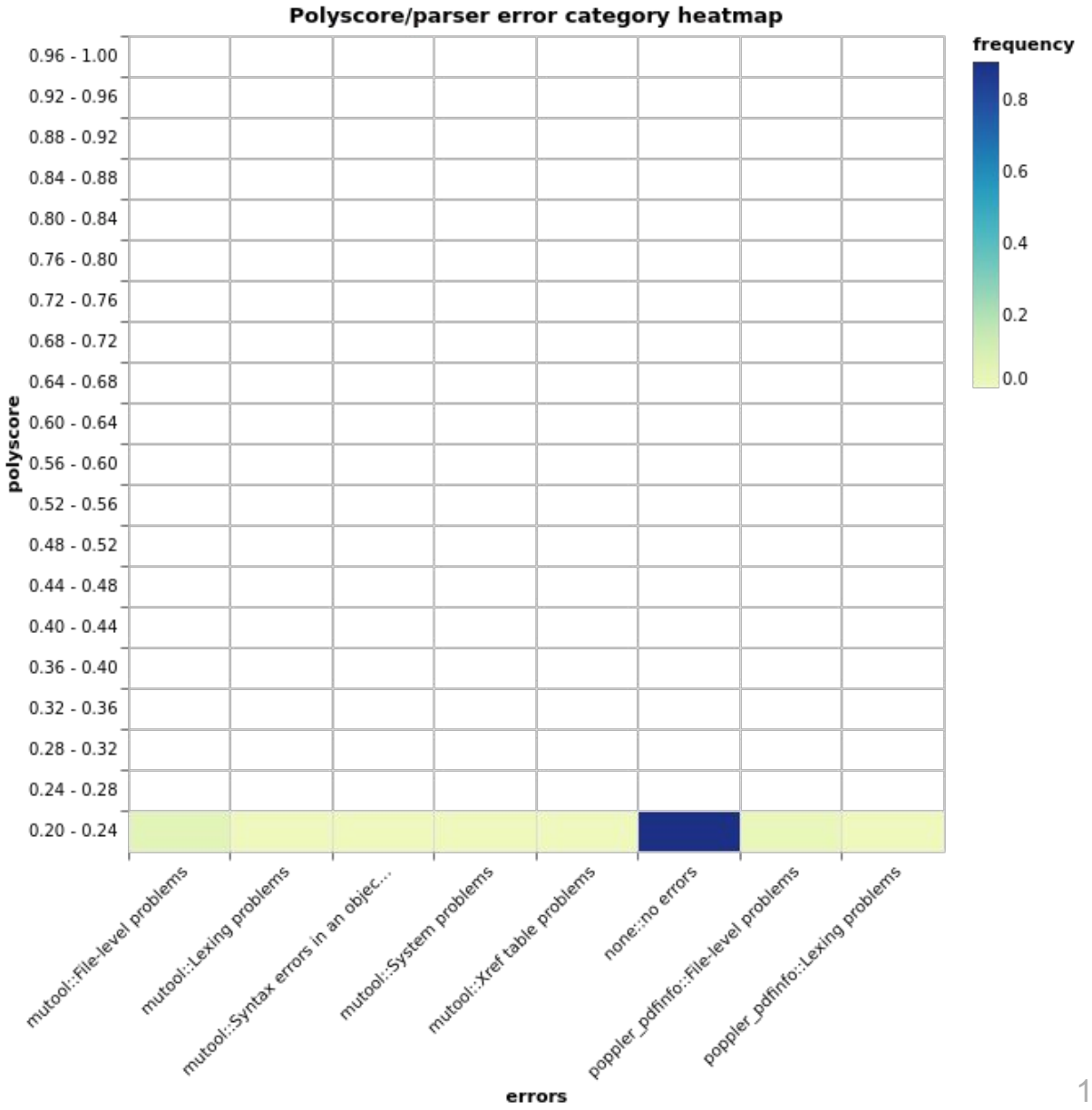
Error Frequencies across different PDF tools

- Most of these files produce no errors
- Poppler errors overlap with Mutool, but Mutool produces far more errors

Error Message	Number of PDF files
No errors	1880
mutool::File-level problems	120
poppler::File-level problems	58
mutool::Xref table	4
mutool::Syntax errors	3
mutool::Lexing problems	1
mutool::System problems	1
poppler::Lexing problems	1
Total Files	2068

Running GovDocs files through PolySwarm





Live Hunt Mode

- Actively scanning the PolySwarm API over several days for PDF files
- 58906 files extracted from PolySwarm
  - Only 22 files were well-formed across all our PDF tools
  - A vast number of these errors were produced by Caradoc
- After extracting the score and labels from PolySwarm, each file is also run through our set of PDF tools

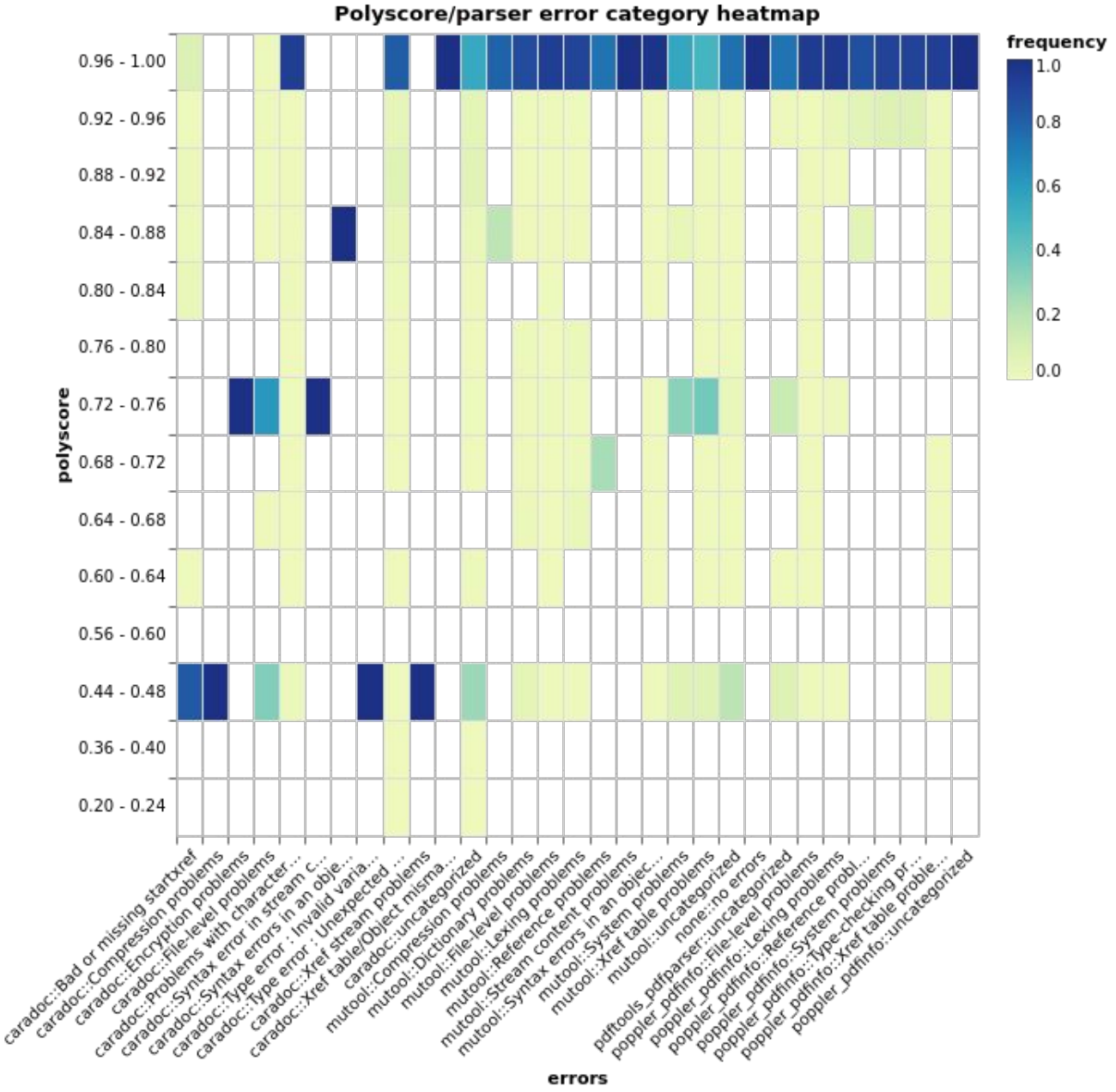
Files by malware category

- Some files may have multiple labels assigned to them
- We explored the top five labels in more detail in the paper
  - Trojans and Cryptominers discussed in more detail in this talk

Malware Category	Count
Trojan	28807
Mass Mailer	25950
Security Assessment Tool	9046
Virus	337
Cryptominer	245
Downloader	79
Prepender	45
Exploit	43
Worm	37
Nonmalware	30
Backdoor	25
Greyware	16
Browser Modifier	15
Dropper	8
CVE	4
Keylogger	4
Password Stealer	4
Injector	3
Adware	2
Spyware	2
Clicker	1
Bot	1
Banker	1

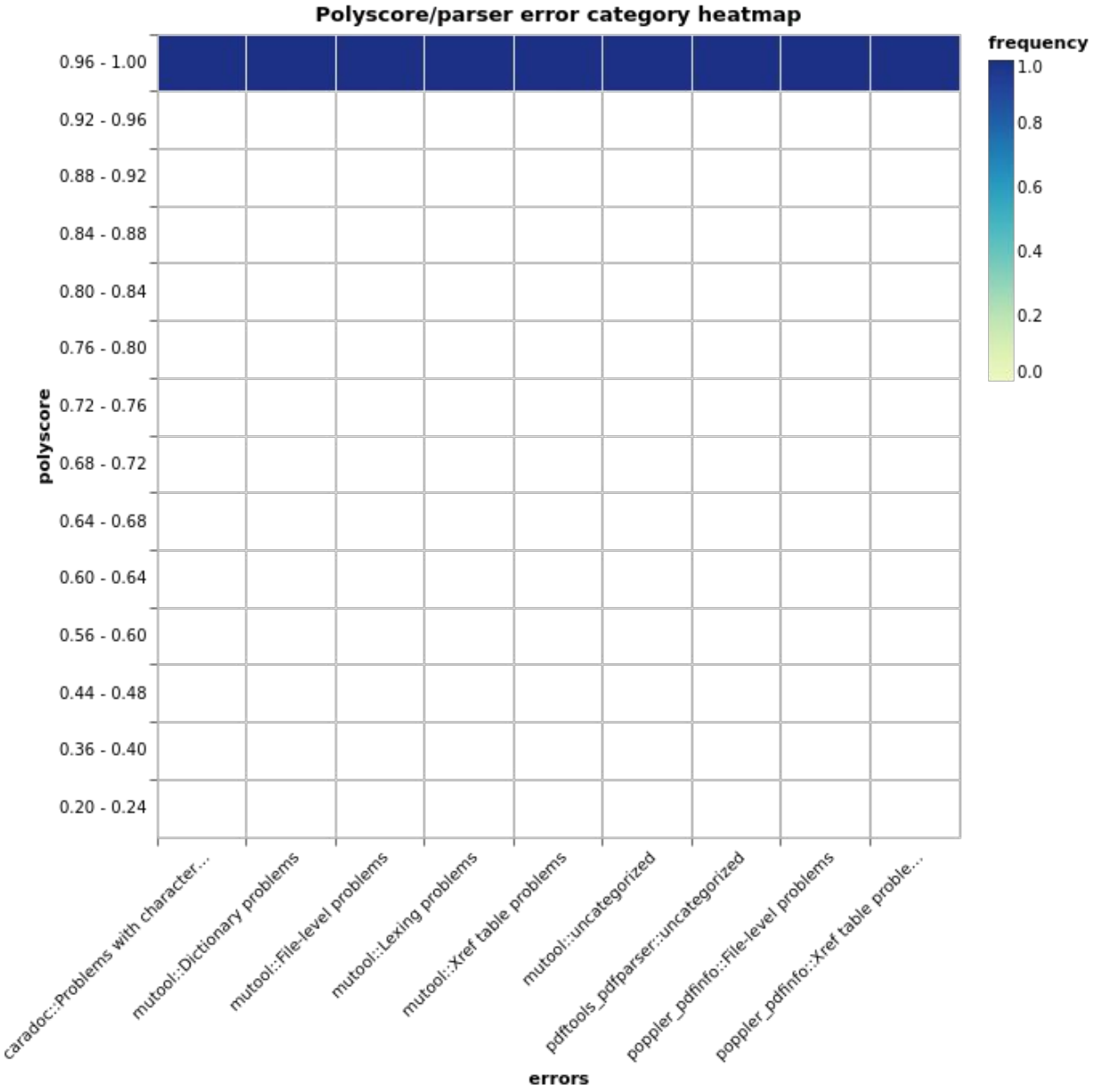
Trojan

- Several errors found by Caradoc tend to be in benign files
- Except a few of the error classes (uncategorized and Xref Table problems in both Poppler and Mutool), Mutool and Poppler find malicious PDFs far more frequently



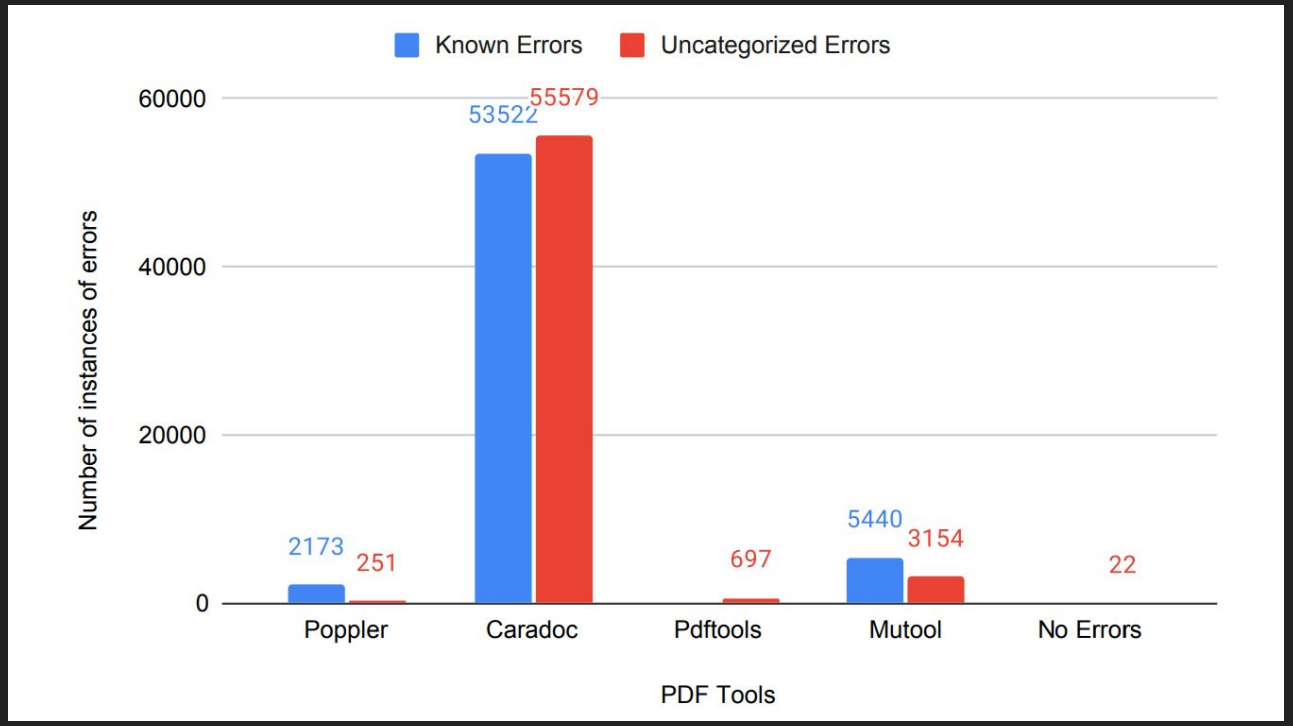
Cryptominer

- File-level and Xref table problems most prevalent in these files
- These files contain the header and %%EOF strings, but do not contain Xref tables or objects within the file
- PE32+ executables built for Windows



Categorizing errors from the error ontology

- Caradoc produces a lot of errors that remain uncategorized
- The percentage of errors unrecognized in Poppler is the lowest among the tools



Outline

- Design Space Overview
- PolyDoc Design
- Findings
  - Baseline experiments
  - PolySwarm experiments
- **Current and Future Directions**

Ongoing and Future Work

- Predicting PolyScore and Malware classes based on error messages from various PDF tools
  - Analyzed a further 7700 files from PolySwarm
  - “Sir-parse-a-lot” is in progress to be integrated into the PolySwarm network as an additional engine
  - Uses patterns from previously seen PolySwarm and GovDocs data to predict malice
- Improving the PDF Error Ontology
- Format-Aware Tracing: Still a challenge
  - One-size-fits-all solutions do not work for different complex data formats: tools need to be adapted extensively

Q & A
Thank you!

prashant.anantharaman@narfindustries.com