

Forum:[Hack Forums](#)

Thread: [PDF Extractor Your PDF Osint tool](#)

HerrWinston - September 13, 2025, 6:50 pm

PDFextractor - The Osint Tool designed to make your life easier!

Look at the bottom for the source code to make it YOUR tool!

This tool scans PDF files for interesting artifacts often hidden inside documents:

URLs

Emails

IP addresses

Obfuscated links

Domains

It generates a console summary + saves a structured JSON report.

Useful for OSINT or malware analysis.

Spoiler ([Click to View](#))

Spoiler ([Click to View](#))

How it works:

Uses pdfminer.six to extract visible text.

Uses PyPDF2 to pull annotation links.

Reads raw byte data (latin1 decode) to catch hidden strings.

Regex patterns detect URLs, emails, IPs, and simple obfuscations.

Domains are normalized with tldextract.

Dont wanna mess with python? No problem!

The Solution is the .exe file you can purchase below. (If you dont have the bytes for it, either reply down here or send me a PM)

You will want to add it to PATH so lets make a little tutorial on how to do this fully in your terminal!

Why would you want it?

By adding it to your path you dont need to write the whole path down everytime when executing a command for example:

```
path\to\pdfextractor -i
```

By adding it you can write:

```
pdfextractor -i
```

So how does that work? (Windows Tutorial)

1. First type: Code `mkdir C:\[desired folder name]` to generate a folder for the tool!
2. Now lets move the .exe in there with: Code `path\to\pdfextractor.exe C:\[desired folder name]`
3. The most important part adding it to the path: Code

```
$add = "C:\Tools"
$up = [Environment]::GetEnvironmentVariable('Path','User')
if (-not ($up.Split(';') -contains $add)) {
    [Environment]::SetEnvironmentVariable('Path', ($up + ';' + $add), 'User')
    Write-Host "C:\Tools added to PATH. Restart your terminal."
} else {
    Write-Host "C:\Tools is already in PATH."
}
}
```
4. Restart your Powershell terminal!
5. Test it with: Code `where pdfextractor` this should print: `C:\Tools\pdfextractor.exe`
6. `pdfextractor --info`
If you see the info page its working perfectly!

Example Usage of the .exe:

Code `pdfextractor "C:\Users\You\Documents\sample.pdf" --pretty` Generates `sample_links.json` in the same folder.

The full source code you can put your personal touch to:

Spoiler ([Click to View](#)) Code

```
import json
import os
import argparse
import collections
from pdfminer.high_level import extract_text
import PyPDF2
import tldextract
```

```
url = re.compile(r'\b(?:https?://|http://|www\.)[^\s<>"\']]+\b', re.I)
email = re.compile(r'[\w\.-]+@[\w\.-]+\.\w{2,}', re.I)
ip = re.compile(r'\b(?:\d{1,3}\.){3}\d{1,3}\b')
```

```
OBFUS_HXXP = re.compile(r'\bhxxps?:\W[^\s<>"\']]+\b', re.I)
OBFUS_HTTP_SQUARE = re.compile(r'http[:\W[^\s<>"\']]+\b', re.I)
```

```
def normalize_domain(host: str) -> str:
    host = host.strip().lower()
    host = re.sub(r'^https?://', '', host)
    host = host.split('/')[0].split(':')[0]
    te = tldextract.extract(host)
    if te.domain and te.suffix:
        return f'{te.domain}.{te.suffix}'
    return host
```

```
def textpdf(path: str):
    try:
        return extract_text(path) or ""
    except Exception:
        return ""
```

```
def annotations(path: str):

    found = set()
    try:
        with open(path, 'rb') as fh:
            reader = PyPDF2.PdfReader(fh)
            for p in reader.pages:
                try:
                    if '/Annots' in p:
                        for a in p['/Annots']:
                            aobj = a.get_object()
                            if '/A' in aobj and '/URI' in aobj['/A']:
                                uri = aobj['/A']['/URI']
                                if isinstance(uri, str):
                                    found.add(uri)
                            if '/URI' in aobj:
                                u = aobj.get('/URI')
                                if u:
                                    found.add(u)
                except Exception:
                    continue
    except Exception:
        pass
    return sorted(found)
```

```
def rawtext(path: str) -> str:
    try:
        with open(path, 'rb') as fh:
            data = fh.read()
            return data.decode('latin1', errors='ignore')
    except Exception:
        return ""
```

```
def analyze_pdf(path: str):
    report = {
        "file": path,
```

```

"urls": [],
"emails": [],
"annotations": [],
"obfuscated": [],
"ips": [],
"domains": {}
}

```

```

text = textpdf(path)
raw = rawtext(path)
annots = annotations(path)

```

```

urls = set(url.findall(text)) | set(url.findall(raw))
for a in annots:
    urls.update(url.findall(a))

```

```

mails = set(email.findall(text)) | set(email.findall(raw))
for a in annots:
    mails.update(email.findall(a))

```

```

obfus = set()
obfus.update(OBFUS_HXXP.findall(text))
obfus.update(OBFUS_HXXP.findall(raw))
obfus.update(OBFUS_HTTP_SQUARE.findall(text))
obfus.update(OBFUS_HTTP_SQUARE.findall(raw))

```

```

ips = set(ip.findall(text)) | set(ip.findall(raw))

```

```

domain_map = {}
for u in urls:
    clean = u.strip('.,;:()[]\'"')
    host = clean
    if clean.lower().startswith('http'):
        host = re.sub(r'^https?://', '', clean, flags=re.I).split('/')[0]
    elif clean.lower().startswith('www.'):
        host = clean.split('/')[0]
    domain = normalize_domain(host)
    domain_map.setdefault(domain, set()).add(clean)

```

```

report['urls'] = sorted(urls)
report['emails'] = sorted(mails)
report['annotations'] = sorted(annots)
report['obfuscated'] = sorted(obfus)
report['ips'] = sorted(ips)
report['domains'] = {d: {"examples": sorted(list(s))[:8]} for d, s in domain_map.items()}

```

```

return report

```

```

# CLI
def cli():
    p = argparse.ArgumentParser(

```

```

description="PDF Link Extractor (mini) — Text + Annotationen + simple Obfuscation
detection")
p.add_argument("file", nargs="?", help="Path to PDF")
p.add_argument("--out", "-o", help="JSON-outputpath (default: <basename>_links.json)")
p.add_argument("--pretty", action="store_true", help="Prettier Version of JSON output")
p.add_argument("--info", "-i", action="store_true", help="Output General information of tool")
args = p.parse_args()

if args.info:
    print("PDF File Link Extractor v1\nFeatures: Text + Annotations + Obfuscation
Detection\nOriginal Autor: HerrWinston on HackForum")
    return

if not args.file:
    p.error("the following argument is required: file (unless --info is used)")

path = args.file
if not os.path.exists(path):
    print("File Not Found:", path)
    return

rpt = analyze_pdf(path)

# Output
print(f"[+] File: {rpt['file']}")
print(f"-> Found URLs: {len(rpt['urls'])}")
for u in rpt['urls'][:50]:
    print("  ", u)
if len(rpt['urls']) > 50:
    print("  ... ({} more)".format(len(rpt['urls']) - 50))

print(f"\n -> Annotation-Links: {len(rpt['annotations'])}")
for anos in rpt['annotations'][:30]:
    print("  ", anos)

print(f"\n -> Emails: {len(rpt['emails'])}")
for e in rpt['emails'][:30]:
    print("  ", e)

print(f"\n -> Obfuscated Hits: {len(rpt['obfuscated'])}")
for o in rpt['obfuscated'][:30]:
    print("  ", o)

print(f"\n -> IPs: {len(rpt['ips'])} (Example: {rpt['ips'][:6]})")

print(f"\n -> Domains ({len(rpt['domains'])}):")
for d, info in list(rpt['domains'].items())[:40]:
    print(f"  - {d} examples: {info['examples'][:3]}")

out = args.out or os.path.splitext(os.path.basename(path))[0] + "_links.json"
try:
    with open(out, 'w', encoding='utf-8') as fh:
        if args.pretty:

```

```
        json.dump(rpt, fh, indent=2, ensure_ascii=False)
    else:
        json.dump(rpt, fh, ensure_ascii=False)
    print("\nReport saved:", out)
except Exception as didntwork:
    print(f"Could not write Output: {didntwork}")

if __name__ == "__main__":
    cli()
```

This thread is the entry for the amazing contest from [7](#) , caring about the community and getting HQ content on this Forum again, thank you man!

itcorp - September 13, 2025, 7:15 pm

I'm glad you joined to this great contest by [7](#) .I can't wait to see the further evolution of this tool from you.Nice share buddy!

CL1B - September 14, 2025, 6:39 am

Yes [HerrWinston](#) Yes

This is good, I hope to see more of your work in the future.

blko57 - September 15, 2025, 8:22 am

thanks bro

Corban8888 - September 16, 2025, 1:47 am

great! Thanks

CL1B - September 24, 2025, 9:13 am

Spoiler ([Click to View](#))

Working fine!