



# PDF Malware

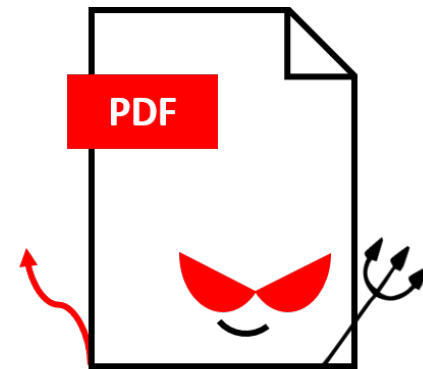
[실습]



**이화여자대학교**  
EWha WOMANS UNIVERSITY

# PDF malware

- PDF document can be malicious !
- The number of detected PDF-based attacks is drastically increasing
  - More than 47,000 new PDF-based attacks were discovered in 2018
  - In 2019, >73k PDF-based attacks in one month
  - PDF malware accounts for 17.42% of newly detected threats in year 2019
- PDF malware is popular because PDF files can be viewed on any device and are easy to create
  - Large attack surface for adversaries



# PDF malware

- Adversaries can inject their own JavaScript code into the PDF document
- Object-based structure
- JS code exploits specific PDF reader's vulnerability to perform malicious actions

```
%PDF-1.3
2 0 obj
<<
/OpenAction << /JS 9 0 R /S /JavaScript >>
/Type /Catalog
/Pages 3 0 R
>>
endobj

9 0 obj
<</Filter /FlateDecode
/Length 2691
>>
stream
x\234\355\_s\3338^N^?\317\247\360\365\341&\2311\33
'V^]5+w\325H\211\345\353\335LVvd[\262\235\313\337\
|^?q\335\212\24\
7\337G\243\321\351\355u\353\240\365\242wx\324\267>
'\323\313h6_\304\311r\265\276\372\317\365\315\355\
\274\337}}\360\342\303\326\345\335*\274\235\257W\2
\337\335\222^F*\217\220c[8\273\336\377\211\376o\26
\311^E\362\277\315\376w\330\377\256\270<
endstream
endobj
```

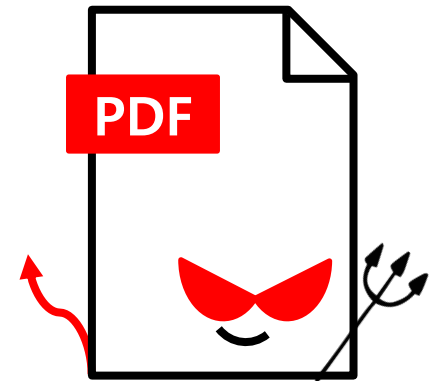
# JavaScript Encoding

- First, adversaries encode malicious JavaScript



# JavaScript Injection

- Then, they inject encoded malicious JavaScript code into PDF structure



# PDF Malware Circulation

- Adversaries spread their malicious PDF documents



Internet



Victim



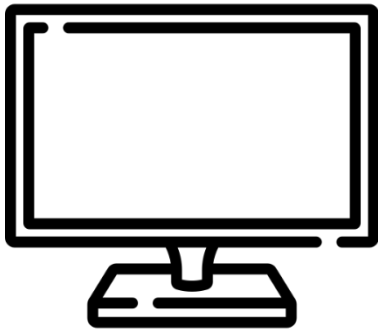
Adversary

# PDF Malware Download

- Adversaries spread their malicious PDF documents



Internet



Victim



Adversary

# PDF Malware Infection

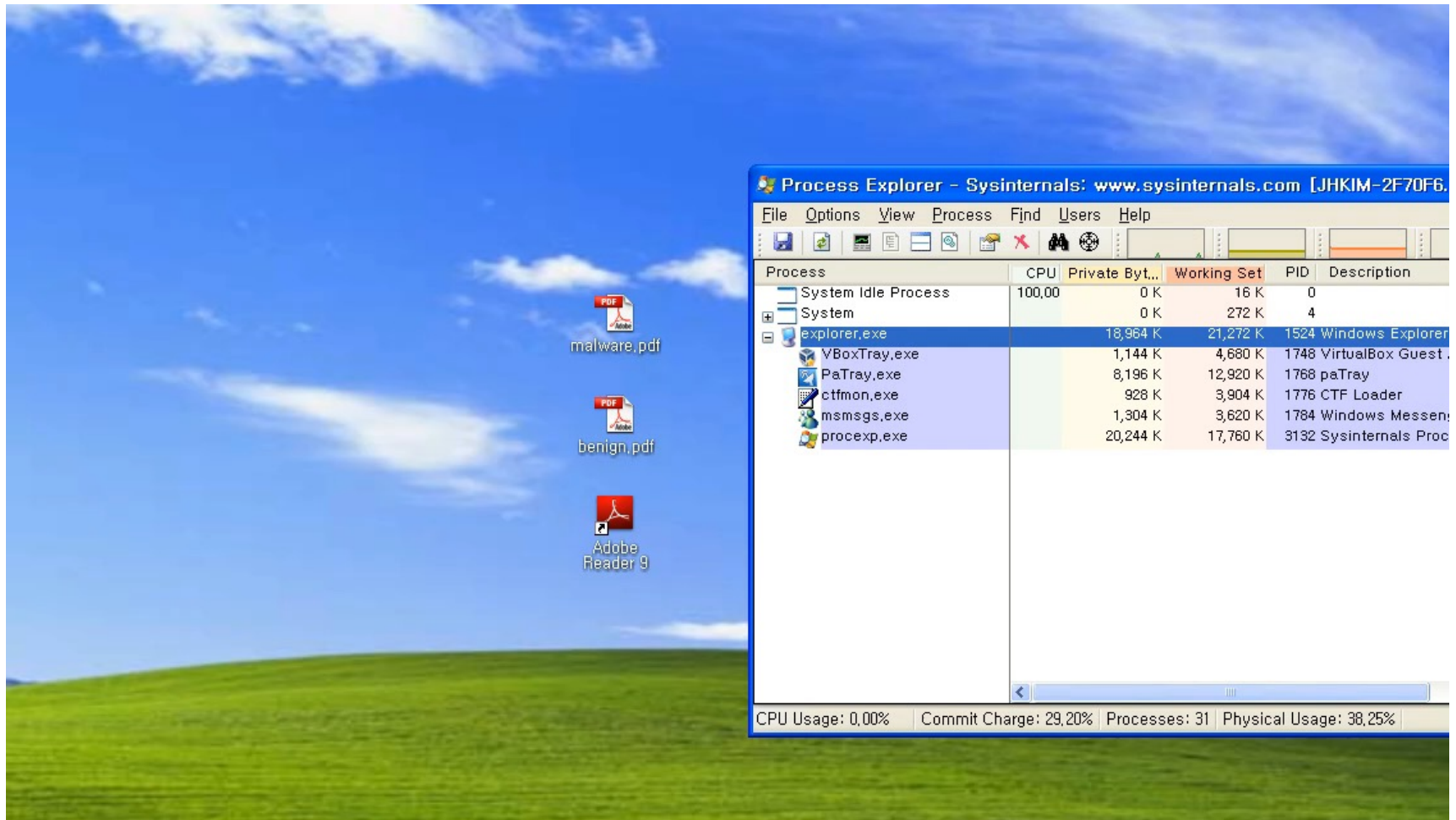
- When the victim opens the PDF malware, ...



Victim

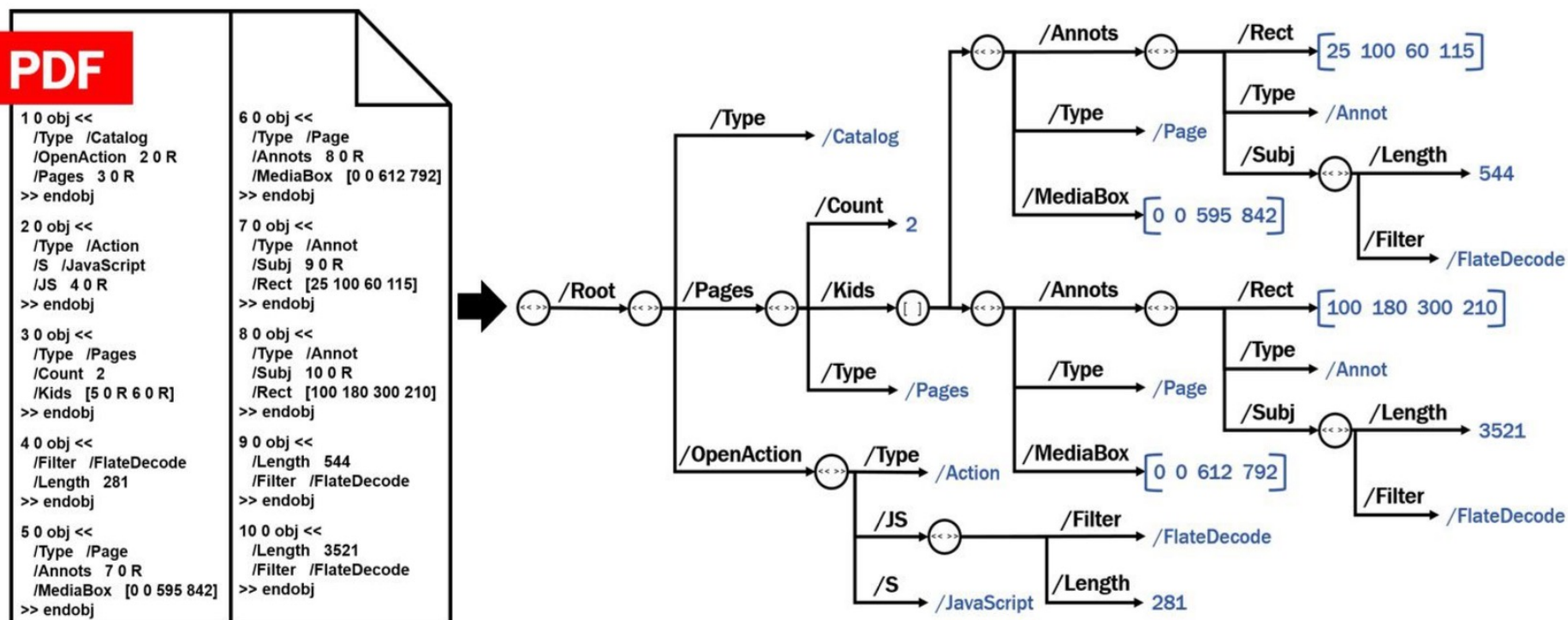


# PDF Malware Infection



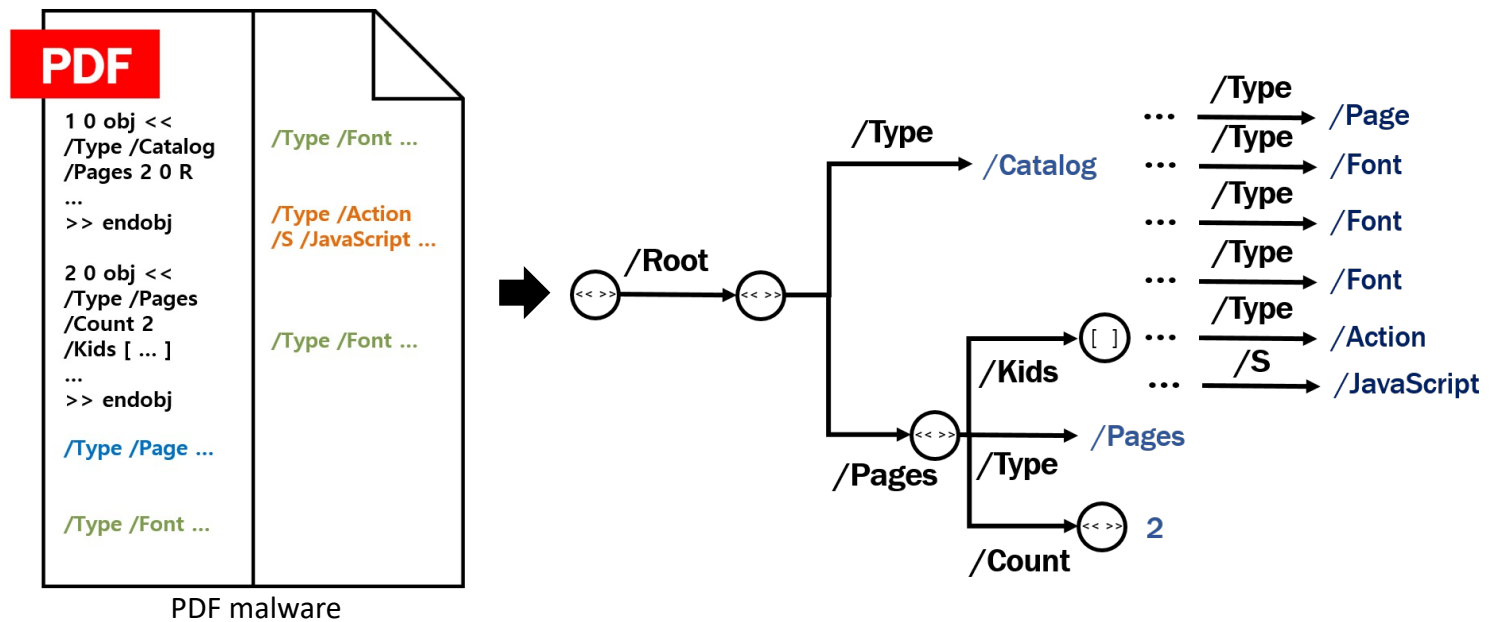
# Hidost: Structure-based Classifier

- Hidost discriminates between malicious and benign files based on the logical structure
- Not a collection of individual features but their relations in the PDF structures.



# Hidost: Structure-based Classifier


- Parse PDF into a structural representation
- The feature extraction identifies the path where **font objects** are located




/Root /Type	/Root/Pages /Type	/Root/Pages /Count	/Root/Pages/... /Type	/Root/Pages/... /S	...
/Catalog	/Pages	2	/Action	/JavaScript	...

# Hidost Implementation

- <https://github.com/srndic/hidost>

 **srndic** Fix whitespace in cacher.cpp.in 950b98d on 20 Sep 2020

hidost	Simplified data type checking in feat_extract.py
src	Fix whitespace in cacher.cpp.in
CMakeLists.txt	pdf2vals now includes both presence and value features
COPYING	Initial commit
INSTALL.rst	Documentation fixes
README.rst	Updated README.md

 README.rst

## HIDOST

---

### Toolset for extracting document structures from PDF and SWF files

---

Copyright 2014 Nedim Srndic, University of Tuebingen [nedim.srndic@uni-tuebingen.de](mailto:nedim.srndic@uni-tuebingen.de)

### Installation and Setup

---

Please consult the `INSTALL.rst` file.

# Hidost Install (1)

- ubuntu 16.04
- `sudo apt-get update`
- `sudo apt-get upgrade`
- `sudo apt-get install git curl vim cmake libboost-all-dev libfontconfig1-dev libopenjpeg-dev libjpeg-dev default-jdk`  
rename

## Hidost Install (2)

- `mkdir ~/hidost`
- `git clone https://github.com/srndic/hidost.git ~/hidost/`
- `cd ~/hidost`
- `git clone https://github.com/srndic/libquickly`
- `cd libquickly`
- `mkdir build`
- `cd build`
- `cmake -DMAKE_TESTS=1 ..`
- `make`
- `sudo make install`
- `cd ../..`

# Hidost Install (3)

- `wget https://github.com/downloads/sporst/SWFREtools/swfretools\_140.zip`
- `mkdir swfretools`
- `unzip swfretools_140.zip -d ~/hidost/swfretools`
- `vi ~/.bashrc`
- `export CLASSPATH=$CLASSPATH:/path_to/hidost/swfretools/dissector.jar` 를 작성
- **path\_to** → 각자 적당한 것으로 수정 필요
- `source ~/.bashrc`
  
- `wget https://poppler.freedesktop.org/poppler-o.18.4.tar.gz`
- `tar -xvzf poppler-0.18.4.tar.gz`
- `cd poppler-0.18.4`
- `./configure`
- `make`
- `sudo make install`
- `sudo cp -r ./poppler /usr/include/`
- `sudo cp -r ./goo /usr/include/poppler/`
- `cd ..`

# Hidost Install (4)

- mkdir build
- cd build
- cmake ..
- make
- sudo make install
- cd ../..
  
- mkdir dataset
- cd dataset
- wget  
[https://www.dropbox.com/sh/t8qak9zwfg45hva/AACRS3dCFAwAAEIPvazxRc1Ka/MALWARE\\_PDF\\_PRE\\_04-2011\\_10982\\_files.zip](https://www.dropbox.com/sh/t8qak9zwfg45hva/AACRS3dCFAwAAEIPvazxRc1Ka/MALWARE_PDF_PRE_04-2011_10982_files.zip)
- unzip MALWARE\_PDF\_PRE\_04-2011\_10982\_files.zip
- mv MALWARE\_PDF\_PRE\_04-2011\_10982\_files ./malicious/
- rm ~/dataset/malicious/log.txt
- cd malicious
- rename 's/\$/.pdf/' \*
- cd ..
  
- wget  
[https://www.dropbox.com/sh/t8qak9zwfg45hva/AABcJgh7iwAOjlgD2uM7Hj92a/CLEAN\\_PDF\\_9000\\_files.zip](https://www.dropbox.com/sh/t8qak9zwfg45hva/AABcJgh7iwAOjlgD2uM7Hj92a/CLEAN_PDF_9000_files.zip)
- unzip CLEAN\_PDF\_9000\_files.zip
- mv CLEAN\_PDF\_9000\_files ./benign



# Hidost Usage (Overview)

1. Prepare the files `bpdfs.txt` and `mpdfs.txt`.
2. Position in the directory where Hidost has been built, as detailed in `INSTALL.rst`:

```
cd build/
```

3. In order to avoid having to extract tree structures from PDF files multiple times, we will cache the extracted structures in cache directories, for benign and malicious files separately:

```
mkdir cache-ben cache-mal
./src/cacher -i bpdfs.txt --compact --values -c cache-ben/ \
-t10 -m256
./src/cacher -i mpdfs.txt --compact --values -c cache-mal/ \
-t10 -m256
```

We will need the absolute paths of all non-empty cached PDF structures in the following steps:

```
find $PWD/cache-ben -name '*.pdf' -not -empty >cached-bpdfs.txt
find $PWD/cache-mal -name '*.pdf' -not -empty >cached-mpdfs.txt
cat cached-bpdfs.txt cached-mpdfs.txt >cached-pdfs.txt
```

4. Now we will count in how many PDF files each of the PDF structural paths occur:

```
./src/pathcount -i cached-pdfs.txt -o pathcounts.bin
```

5. The next step is feature selection. We will only take into account structural paths present in at least 1,000 PDF files in our dataset:

```
./src/feat-select -i pathcounts.bin -o features.nppf -m1000
```

6. Finally, we will extract the selected features from all files and store the result in the output file `data.libsvm`:

```
./src/feat-extract -b cached-bpdfs.txt -m cached-mpdfs.txt \
-f features.nppf --values -o data.libsvm
```

The output file `data.libsvm` can now be used for learning and classification.

# Hidost Usage (1) bpdfs.txt & mpdfs.txt

- bpdfs.txt with “absolute path (full path)” of benign pdfs
- Mpdfs.txt with “absolute path (full path)” of malicious pdfs
- `ls -d $PWD/* > ~/hidost/build/bpdfs.txt`
- `cd ~/dataset/malicious`
- `ls -d $PWD/* > ~/hidost/build/mpdfs.txt`

- bpdfs.txt

```
/home/pdfmal/dataset/benign/02eounrel.pdf  
/home/pdfmal/dataset/benign/02frrltr.pdf  
/home/pdfmal/dataset/benign/02govbnd.pdf  
/home/pdfmal/dataset/benign/02solp.pdf  
/home/pdfmal/dataset/benign/030.pdf  
/home/pdfmal/dataset/benign/031mandelson.pdf  
/home/pdfmal/dataset/benign/032.pdf  
/home/pdfmal/dataset/benign/033.pdf
```

- mpdfs.txt

```
/home/pdfmal/dataset/malicious/0000fc3840119cade9fb2dae9576cafd3be7f923.pdf  
/home/pdfmal/dataset/malicious/00060c6f8c34a29e9aa4b87e1f00f87b38389a27.pdf  
/home/pdfmal/dataset/malicious/000fbadb686b99b8fc1007c773f05b491a6c87da.pdf  
/home/pdfmal/dataset/malicious/001d92fc29146e01e0ffa619e5dbf23067f1e814.pdf  
/home/pdfmal/dataset/malicious/0025fd602ce1a67e740403dfec8f1d80ed5ceec0.pdf  
/home/pdfmal/dataset/malicious/002779aac17e85baf9d2dab135cae9bac7da3156.pdf  
/home/pdfmal/dataset/malicious/003d7bff3d40449cfd6b920bcacb2c3445141748.pdf  
/home/pdfmal/dataset/malicious/003fddfdf2c321cbc9aa58ec41ed8b9b13f9394e.pdf
```

## Hidost Usage (2) cache-ben & cache-mal

- sudo reboot
- cd ~/hidost/build/
- mkdir cache-ben cache-mal
- ./src/cacher -i bpdfs.txt --compact --values -c cache-ben/ -t10 -m256
- ./src/cacher -i mpdfs.txt --compact --values -c cache-mal/ -t10 -m256
  - compact : perform feature compaction
  - values : use values instead of presence as features
  - c : where to cache
  - t : limit the CPU time of child processes in seconds
  - m : limit the virtual memory of child processes in

```
pdfmal@pdfmal-VirtualBox:~/hidost/build$ ./src/cacher -i bpdfs.txt --compact --values -c cache-ben/ -t10 -m256
ThreadPool running with 1 threads.
8900
8800
8700
8600
8500
8400
8300
8200
```

## Hidost Usage (3) cached-pdfs.txt

- `find $PWD/cache-ben -name '*.pdf' -not -empty > cached-bpdfs.txt`
- `find $PWD/cache-mal -name '*.pdf' -not -empty > cached-mpdfs.txt`
- `cat cached-bpdfs.txt cached-mpdfs.txt > cached-pdfs.txt`

```
/home/pdfmal/hidost/build/cache-ben/home/pdfmal/dataset/benign/builsear.pdf
/home/pdfmal/hidost/build/cache-ben/home/pdfmal/dataset/benign/a-04-46.pdf
/home/pdfmal/hidost/build/cache-ben/home/pdfmal/dataset/benign/f5713sc.pdf
/home/pdfmal/hidost/build/cache-ben/home/pdfmal/dataset/benign/cantonsd.pdf
/home/pdfmal/hidost/build/cache-ben/home/pdfmal/dataset/benign/colknow.pdf
/home/pdfmal/hidost/build/cache-ben/home/pdfmal/dataset/benign/ics-pia.pdf
/home/pdfmal/hidost/build/cache-ben/home/pdfmal/dataset/benign/67fr78176.pdf
/home/pdfmal/hidost/build/cache-ben/home/pdfmal/dataset/benign/ann97-52.pdf
/home/pdfmal/hidost/build/cache-ben/home/pdfmal/dataset/benign/095001f.pdf
/home/pdfmal/hidost/build/cache-ben/home/pdfmal/dataset/benign/f8554.pdf
/home/pdfmal/hidost/build/cache-ben/home/pdfmal/dataset/benign/adkeeper.pdf
/home/pdfmal/hidost/build/cache-ben/home/pdfmal/dataset/benign/armedzilla.pdf
/home/pdfmal/hidost/build/cache-ben/home/pdfmal/dataset/benign/n-11-66.pdf
/home/pdfmal/hidost/build/cache-ben/home/pdfmal/dataset/benign/artechmed.pdf
/home/pdfmal/hidost/build/cache-ben/home/pdfmal/dataset/benign/chromlif.pdf
/home/pdfmal/hidost/build/cache-ben/home/pdfmal/dataset/benign/broadclp.pdf
/home/pdfmal/hidost/build/cache-ben/home/pdfmal/dataset/benign/i1120icd.pdf
/home/pdfmal/hidost/build/cache-ben/home/pdfmal/dataset/benign/aracamer.pdf
/home/pdfmal/hidost/build/cache-ben/home/pdfmal/dataset/benign/i1045.pdf
```



# Hidost Usage (4) pathcounts.bin

- `./src/pathcount -i cached-pdfs.txt -o pathcounts.bin`
  - Count the number of pdfs with each path (feature)
  - Path / number of pdfs found

```
1 <nn>^@^@ 27
2 AA^@^@ 5
3 AA^@DP^@^@ 1
4 AA^@DP^@JS^@^@ 1
5 AA^@DP^@JS^@Filter^@^@ 1
6 AA^@DP^@JS^@Length^@^@ 1
7 AA^@DP^@S^@^@ 1
8 AA^@DS^@^@ 1
9 AA^@WC^@^@ 2
10 AA^@WC^@JS^@^@ 1
11 AA^@WC^@JS^@Filter^@^@ 1
12 AA^@WC^@JS^@Length^@^@ 1
13 AA^@WC^@S^@^@ 1
14 AA^@WP^@^@ 1
15 AA^@WS^@^@ 1
16 AcroForm^@^@ 4520
17 AcroForm^@CO^@^@ 9
18 AcroForm^@CO^@AA^@C^@^@ 9
19 AcroForm^@CO^@AA^@C^@JS^@^@ 9
20 AcroForm^@CO^@AA^@C^@JS^@Filter^@^@ 2
21 AcroForm^@CO^@AA^@C^@JS^@Length^@^@ 2
```

```
3873 ViewerPreferences^@<nn>^@^@ 27
3874 ViewerPreferences^@CenterWindow^@^@ 1
3875 ViewerPreferences^@Direction^@^@ 102
3876 ViewerPreferences^@FitWindow^@^@ 3
3877 ViewerPreferences^@HideMenubar^@^@ 1
3878 ViewerPreferences^@HideToolbar^@^@ 1
3879 ViewerPreferences^@HideWindowUI^@^@ 1
3880 ViewerPreferences^@PageDirection^@^@ 62
3881 ViewerPreferences^@PageDirection^@^@ 1
3882 ViewerPreferences^@Rights^@Document^@^@ 9
3883 ViewerPreferences^@Rights^@Form^@^@ 68
3884 ViewerPreferences^@Rights^@Msg^@^@ 95
3885 ViewerPreferences^@Rights^@RightsID^@^@ 96
3886 ViewerPreferences^@Rights^@TimeOfUbiquitization^@^@ 96
3887 ViewerPreferences^@Rights^@Version^@^@ 96
3888 openAction^@JS^@^@ 21
3889 openAction^@S^@^@ 35
3890 outlines^@^@ 38
3891 outlines^@Count^@^@ 38
3892 outlines^@Type^@^@ 38
```

# Hidost Usage (5) features.nppf

- `./src/feat-select -i pathcounts.bin -o features.nppf -m1000`
  - Select features from all paths with threshold m (found in at least m number of pdfs)
  - m : least number of files that such path is present
  - Change m value for more features

```
1 NPPF^@^@
2 AcroForm^@^@
3 AcroForm^@DA^@^@
4 AcroForm^@DR^@Encoding^@PDFDocEncoding^@^@
5 AcroForm^@DR^@Encoding^@PDFDocEncoding^@Differences^@^@
6 AcroForm^@DR^@Encoding^@PDFDocEncoding^@Type^@^@
7 AcroForm^@DR^@Font^@Name^@^@
8 AcroForm^@DR^@Font^@Name^@BaseFont^@^@
9 AcroForm^@DR^@Font^@Name^@Encoding^@^@
10 AcroForm^@DR^@Font^@Name^@FirstChar^@^@
11 AcroForm^@DR^@Font^@Name^@FontDescriptor^@^@
```

```
268 StructTreeRoot^@ParentTree^@Nums^@Type^@^@
269 StructTreeRoot^@ParentTreeNextKey^@^@
270 StructTreeRoot^@Pg^@^@
271 StructTreeRoot^@RoleMap^@^@
272 StructTreeRoot^@RoleMap^@Name^@^@
273 StructTreeRoot^@S^@^@
274 StructTreeRoot^@T^@^@
275 StructTreeRoot^@Type^@^@
276 Type^@^@
```

# Hidost Usage (6) data.libsvm

- Integers, Real numbers, and Booleans = as they are
- Strings, PDF names, and other non-numeric types = 1

/OpenAction/JS:	alert('Hello!');
/OpenAction/S:	/JavaScript
/Pages/Count:	2
/Pages/Kids/MediaBox:	0 0 612 792 0 0 333 444
/Pages/Kids/Parent:	3 0 R 3 0 R
/Pages/Kids/Resources:	...
/Pages/Kids/Type:	/Page /Page
/Pages/Type:	/Pages
/Type:	/Catalog

Keys Values

/OpenAction/JS:	1.0	1.0
/OpenAction/S:	1.0	1.0
/Pages/Count:	2	2.0
/Pages/Kids/MediaBox:	0 0 612 792 0 0 333 444	166.5
/Pages/Kids/Parent:	1.0 1.0	1.0
/Pages/Kids/Resources:	...	...
/Pages/Kids/Type:	1.0 1.0	1.0
/Pages/Type:	1.0	1.0
/Type:	1.0	1.0

Structural Multimap Feature Vector



# Hidost Usage (6) data.libsvm

- `./src/feat-extract -b cached-bpdfs.txt -m cached-mpdfs.txt -f features.nppf --values -o data.libsvm`
  - benign (0) or malicious (1) / key:value / path of the file
  - key:value can be found in features.nppf
- Extract features from all pdfs with selected features

```
1312 1 57:1 61:1 62:1 63:158 70:1 71:1 72:1 73:0 74:1 79:1 80:1 89:1 91:100 93:
    1 94:1 95:8302 96:1 97:1 101:1 103:612 210:1 275:1 #/home/pdfmal/hidost/bu
    ild/cache-mal/home/pdfmal/dataset/malicious/1e586e6cc74bce577888b2d69b5cae
    3189f308b4.pdf
1313 1 50:1 51:1 52:1 53:1 54:1 55:76530 56:1 73:0 74:1 78:1 79:1 101:1 117:1 1
    91:1 210:1 275:1 #/home/pdfmal/hidost/build/cache-mal/home/pdfmal/dataset/
    malicious/1e5a010886b9bb6ed53262a7ec29b45d067e58a6.pdf
1314 1 57:1 61:1 62:1 63:295 70:1 71:1 78:1 79:1 80:1 89:1 91:250 93:1 94:1 95:
    6004 96:1 97:1 101:1 103:612 210:1 275:1 #/home/pdfmal/hidost/build/cache-
    mal/home/pdfmal/dataset/malicious/1e5bb3fa85fd999cb45925b627bd0940c2cacbf3
    .pdf
1315 1 57:1 61:1 63:3066 70:1 71:1 72:1 73:0 74:1 79:1 80:1 89:1 91:100 93:1 95
    :90630 96:1 97:1 101:1 103:612 210:1 275:1 #/home/pdfmal/hidost/build/cach
    e-mal/home/pdfmal/dataset/malicious/1e655be92bf76ddc5540a83eee255beedd6153
    a5.pdf
```