Demand Driven Information Flow Analysis of WebView in Android Hybrid Apps

Accepted at ISSRE'23@Florence

Jyoti Prakash*
OpenText India

Abhishek Tiwari*
University of Southern Denmark

Christian Hammer
University of Passau

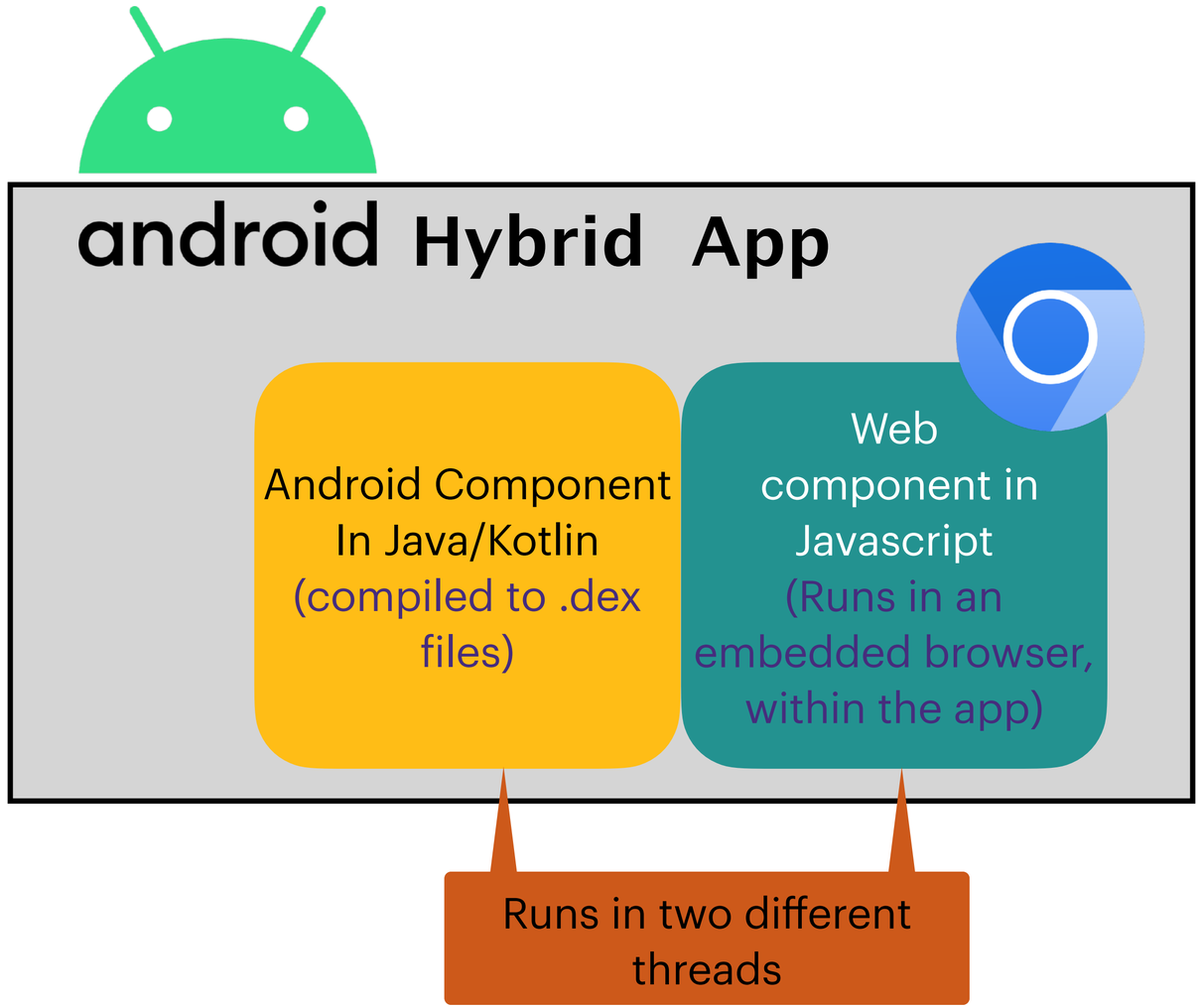
RHPL@FSTTCS'24, IIT Gandhinagar

UNIVERSITÄT PASSAU
Fakultät für Informatik und Mathematik

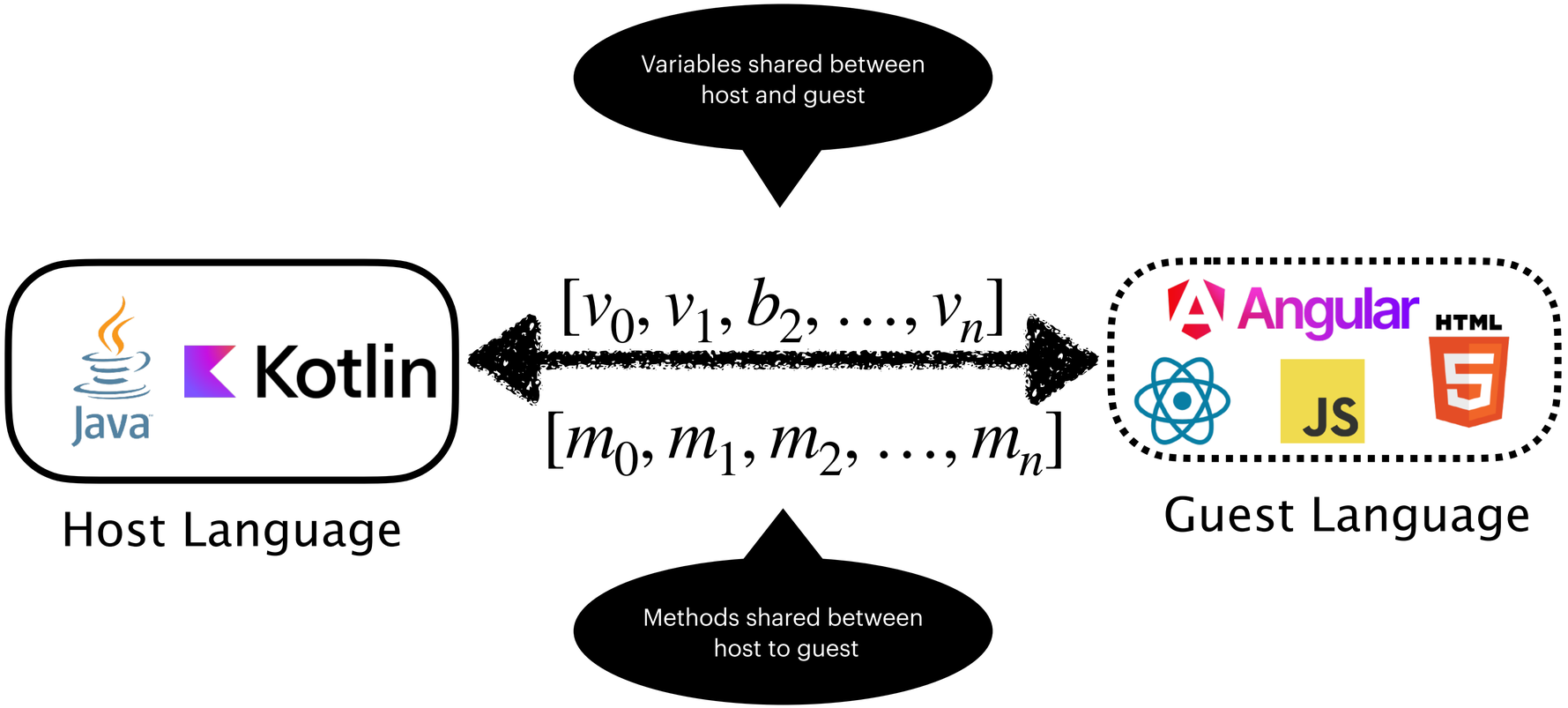
Disclaimer

The work was done when I was affiliated with the University of Passau. The views and opinions expressed in this presentation are solely mine and do not necessarily represent the views, policies, or positions of my employer or any affiliated organisations.

Android Hybrid Apps



WebView Programming Model



Multilingual Programming in Hybrid Apps

android

```
1. foo(WebView myWebView) {
2.   JSObj js = new JSObj(); //obj@iface
3.   myWebView.addJavaScriptInterface(js,"jsobj") //webview
4.   myWebView.evaluateJavascript("set()");
5.   myWebView.loadUrl("javascript:alert('Hello World')");
6.   myWebView.loadUrl("https://www.google.com");
7. }
```

APIs

```
1. class JSObj {
   …

10.    @JavaScriptInterface
11.    public void setValue(Object p) {
12.       this.f = p;
13.    }

200. }
```

Java

```
21. set() {
22.   x = new Object() // obj@x
23.   x.f = JSON.parse(api.getResult())
24.   v = x.f
25.   jsobj.setValue(v)
26. }
```

Javascript

Multilingual Programming in Hybrid Apps


```
1. foo(WebView myWebView) {
2.    JSObj js = new JSObj(); //obj@iface
3.    myWebView.addJavaScriptInterface(js,"jsobj") //webview
4.    myWebView.evaluateJavascript("set()");
5.    myWebView.loadUrl("javascript:alert('Hello World')");
6.    myWebView.loadUrl("https://www.google.com");
7. }
```

```
1. class JSObj {
   …

10.    @JavaScriptInterface
11.    public void setValue(Object p) {
12.       this.f = p;
13.    }

200. }
```

Shared/Bridge methods

Java

```
21. set() {
22.    x = new Object() // obj@x
23.    x.f = JSON.parse(api.getResult())
24.    v = x.f
25.    jsobj.setValue(v)
26. }
```

Javascript

Multilingual Programming in Hybrid Apps

android

Bridge/Interface variable

```
1. foo(WebView myWebView) {
2.   JSObj js = new JSObj(); //obj@iface
3.   myWebView.addJavaScriptInterface(js,"jsobj") //webview
4.   myWebView.evaluateJavascript("set()");
5.   myWebView.loadUrl("javascript:alert('Hello World')");
6.   myWebView.loadUrl("https://www.google.com");
7. }
```

```
1. class JSObj {
   …

10.    @JavaScriptInterface
11.    public void setValue(Object p) {
12.       this.f = p;
13.    }

200. }
```

Java

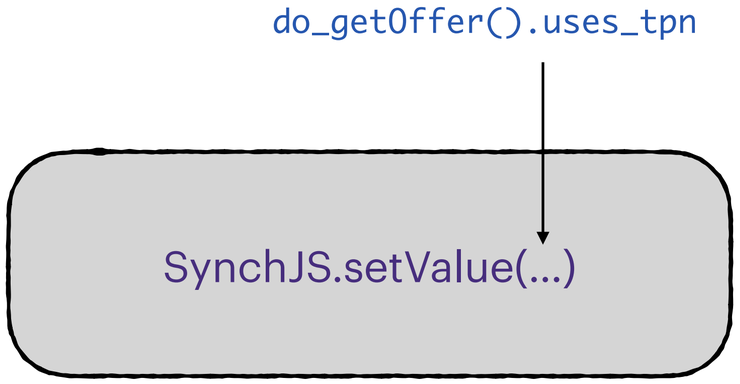
```
21. set() {
22.    x = new Object(); // obj@x
23.    x.f = JSON.parse(api.getResult());
24.    v = x.f;
25.    jsobj.setValue(v)
26. }
```

Javascript

Information Flow Analysis

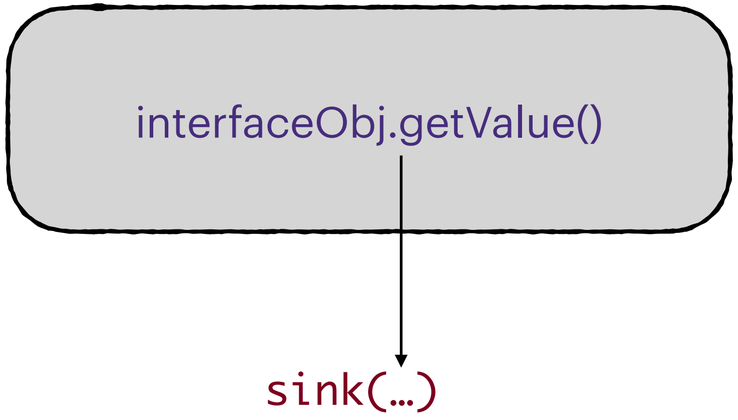
Inter-language Threat Model For Bridge Objects

```
1 javascript :window.SynchJS.setValue((function(){
2     try{
3         return JSON.parse(Sponsorpay.MBE
      .SDKInterface.do_getOffer()).uses_tpn;
4     }catch( js_eval_err ){
5         return false;
6 }})());
```



Integrity Violation


```
1 function set() {
2         var x = interfaceObj.getValue();
3         Sink(x); //New code to Sink secret x
4 }
```



Confidentiality Violation

Peculiarities in WebView Brigde Communication

Flow-Sensitive Update of Bridged Object

```
1. wv.addJavasriptInterface(ifcObj,"ifcObj");
2. ifcObj.g = "publicData";
3. wv.loadUrl("read()");
4. ifcObj.g = "secret";
```

loadUrl is called before line 4

```
1.function read() {
2.    var x = ifcObj.getG();
3.    leak(x);
4.}
5.
```

Fetches the value "secret"

```
1. wv.addJavasriptInterface(ifcObj,"ifcObj");
2. ifcObj.g = "publicData";
3. wv.loadUrl("update()");
4. ifcObj.g = "secret";
```

Fetches the value "1"

```
1.function update() {
2.   ifcObj= {
3.      getG: function () {
4.         return "1";
5.       }
6.   };
7.   var x = ifcObj.getG(); // x = "1"
8.}
```

Overrides the interfaced method getG()

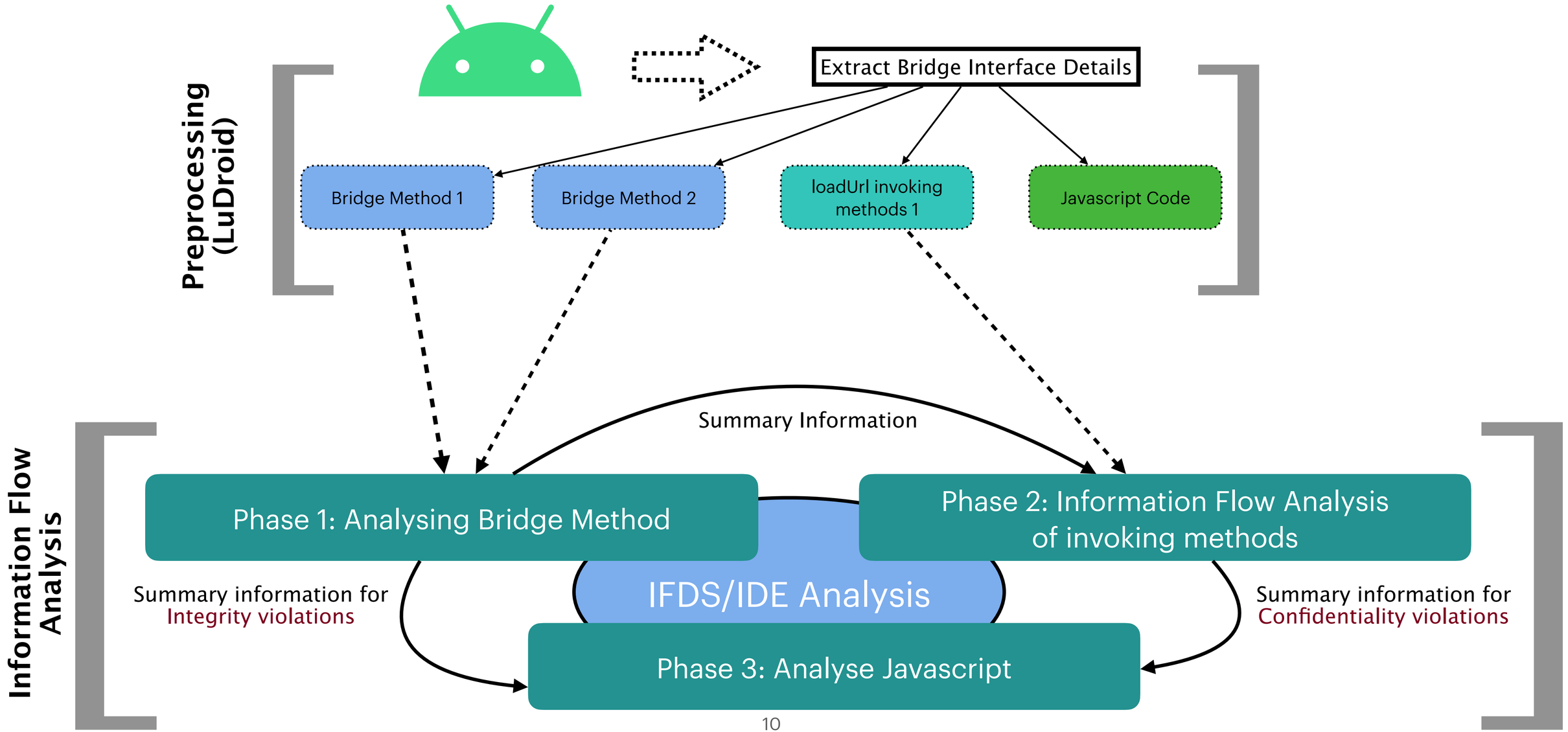
```
1. wv.addJavasriptInterface(ifcObj,"ifcObj");
2. ifcObj.g = "publicData";
3. wv.loadUrl("update()");
4. ifcObj.g = "secret";
```

Deletes the bridge object

```
1.function delete() {
2.   delete ifcObj;
3.   var x = ifcObj.getG();
4.}
```

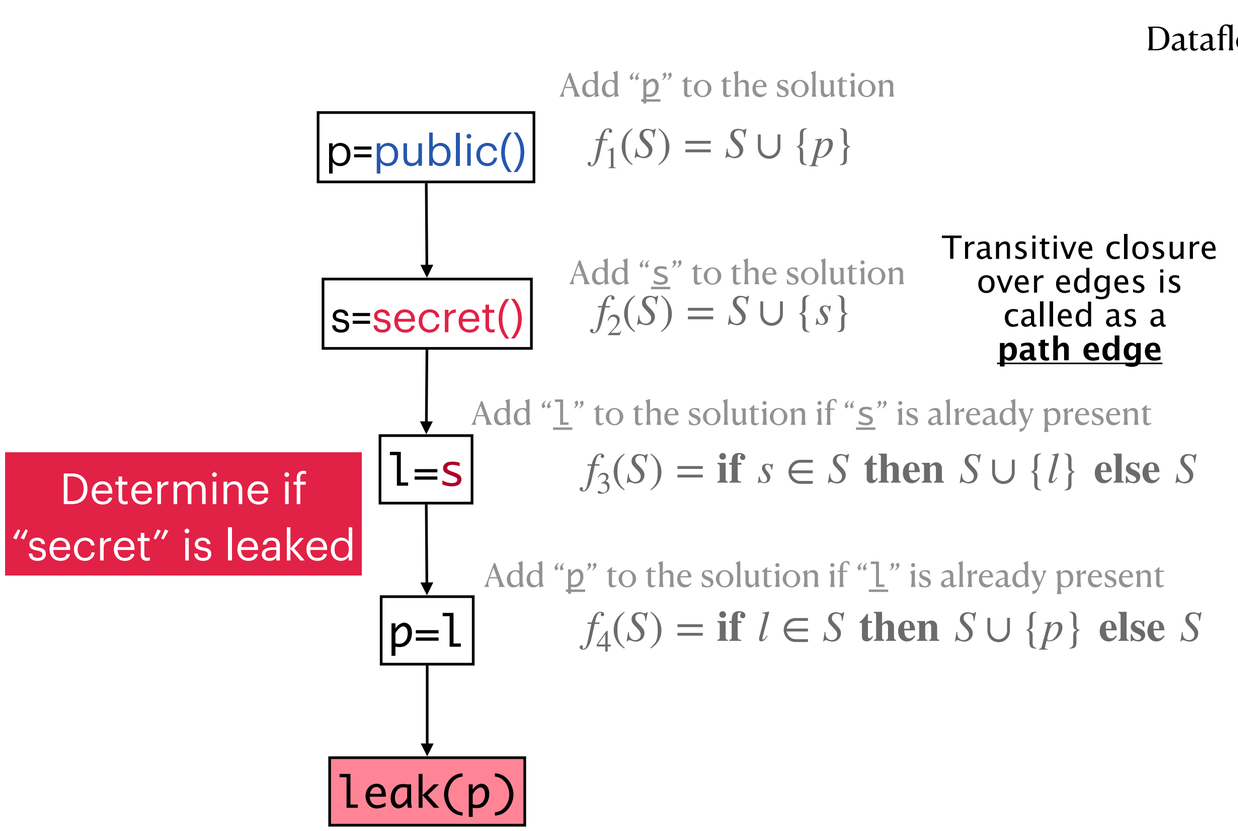
Ignores the delete operations

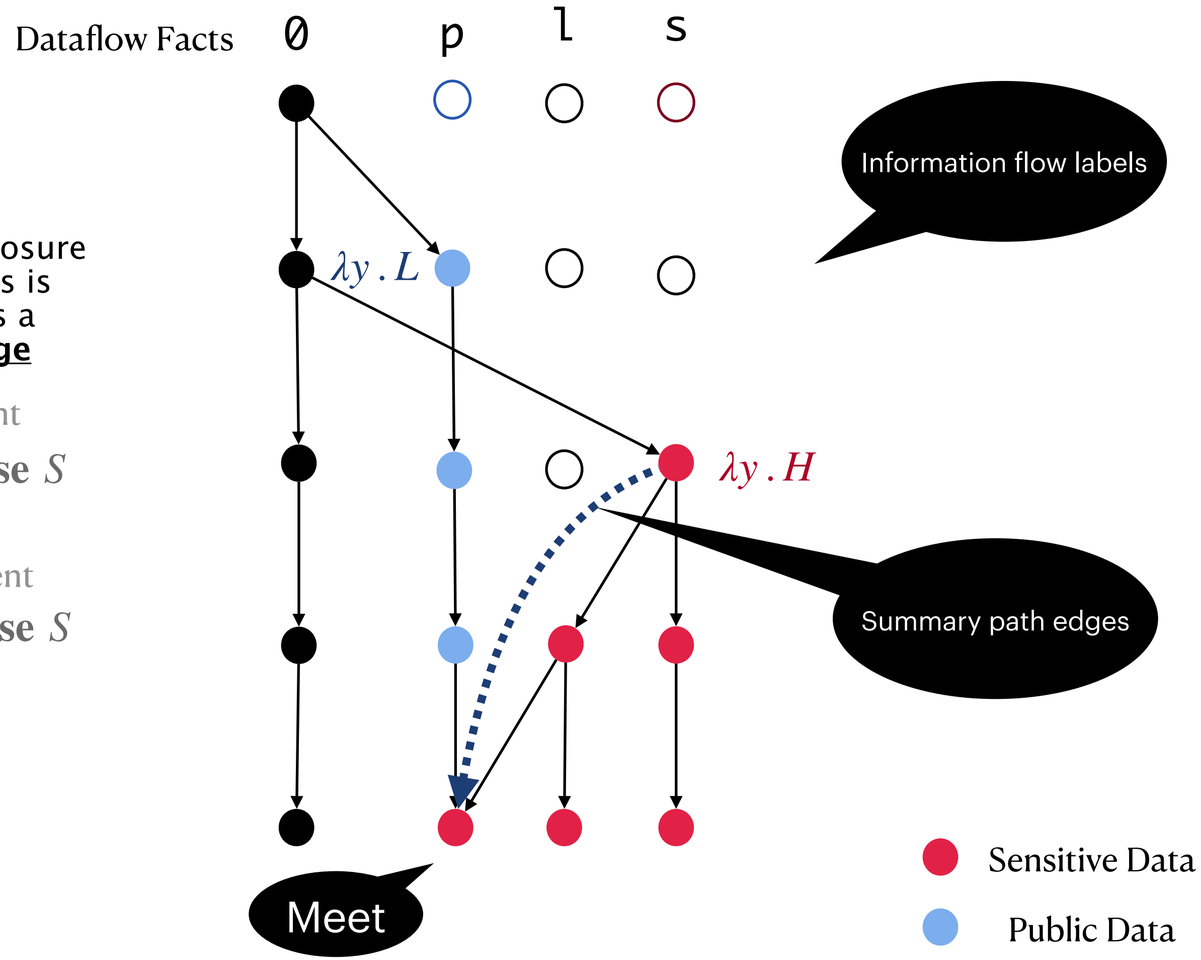
IWanDroid: Information Flow Analysis on WebView



IFDS/IDE Analysis

Computes certain dataflow analyses problem with graph reachability



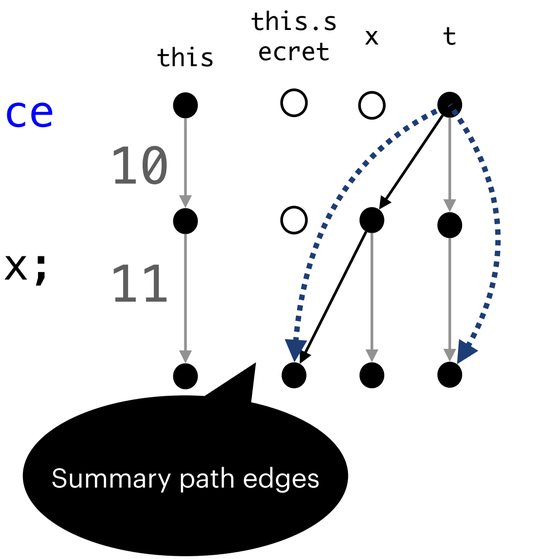


Transfer functions to compute reachability

Computes Information Flow Labels

Analysis: Phase 1
Identify variables which are likely to influenced

```
   @JavaScriptInterface
 9 set(Object t) {
10     var x = t;
11     this.secret = x;
   }
```



Transitive closure over edges is called as a **path edge**

Forward analysis to discover
the set of access-paths modified
by the input parameters to the function

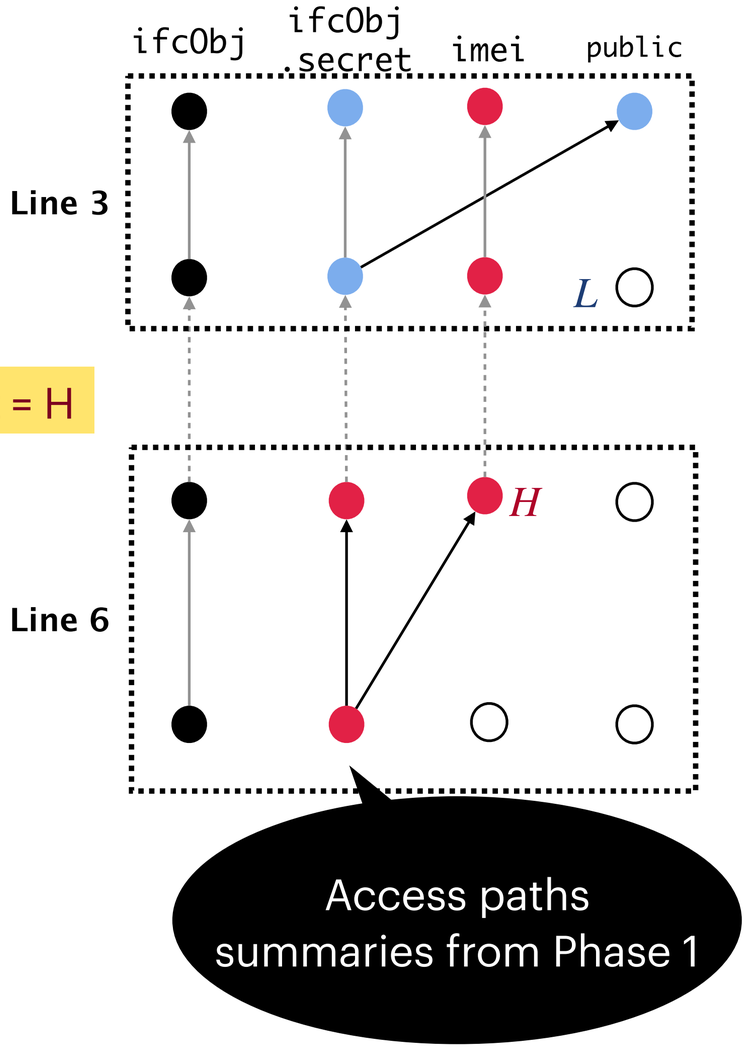
Analysis: Phase 2
Analyse Invoking Methods: Backward Taint Analysis

```
1. foo() {
2.    BridgeObj ifcObj = new BridgeObj(this)
3.    ifcObj.set("public");
4.    webView.evaluateJavascript("get()");
5.    var imei = getImei();
6.    ifcObj.set(imei);
7. }
```

Access paths from Phase-1

```
1. foo() {
2.    BridgeObj ifcObj = new BridgeObj(this)
3.    ifcObj.secret = "public";
4.    webView.evaluateJavascript("get()");
5.    var imei = getImei();
6.    ifcObj.secret = imei;
7.  }
```

ifcObj.secret = H



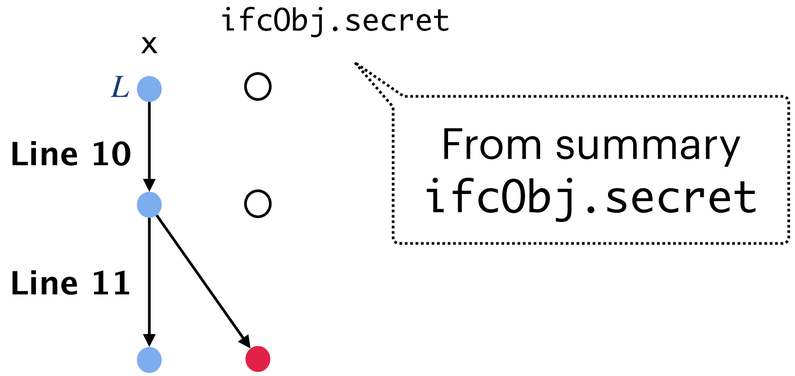
Backward Taint Analysis using access graphs from Phase 1

Analysis: Phase 3
Analyse Javascript Analysis : Forward Taint Analysis

```
 9 function function1() {
10     var x = api().getResult;
11     ifcObj.set(x); //integrity-violation
12 }
```

One of the parameters to the bridged interface is public?

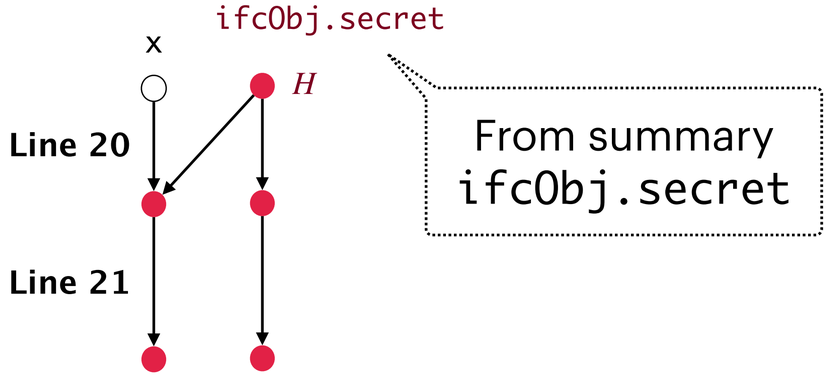
ifcObj.secret = H




```
19 function function2() {
20     var x = ifcObj.get();
21     leak(x); //confidentiality-violation
22 }
```

Replace the summary information use it in taint analysis

```
@JavaScriptInterface
get() {
    return this.secret;
}
```



Analysis: Phase 3
Analyse Javascript Analysis : Forward Taint Analysis

```
1. wv.addJavasriptInterface(ifcObj,"ifcObj");
2. ifcObj.g = "publicData";
3. wv.loadUrl("update()");
4. ifcObj.g = "secret";
```

```
1.function update() {
2.   ifcObj= {
3.      getG: function () {
4.         return "1";
5.       }
6.   };
7.   var x = ifcObj.getG(); // x = "1"
8.}
```

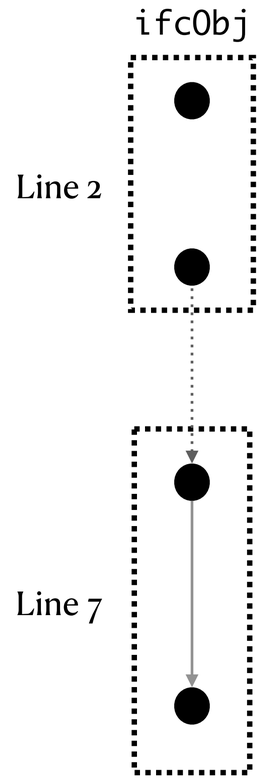
Overrides the interfaced method getG()

Fetches the value "1"

Algorithm 4 IDE_{js}

Require: m_{js}: Bridge Javascript method
Require: *IfcValue*: IfcValue for bridged interfaces $PathWorkList \leftarrow \emptyset$
1: **procedure** ANALYZEJAVASCRIPT
2: **for** every v in Variables(m_{js}) **do**
3: INITIALIZE($\langle m_{js}, v\rangle \rightarrow \langle m_{js}, v\rangle, id$)
4: **end for**PROPAGATE($flowfunctions_{js}$)
5: **end procedure**

6: **procedure** RESOLVETARGET($\langle s, f\rangle$, $PathWorkList$) $targetfunctions \leftarrow \emptyset$
7: **for** path $\langle s_0, f_0\rangle \rightarrow \langle s, f\rangle \in PathWorkList$ **do**
8: **if** $type(f_0)$ is a function **then**
9: $targetFunctions \leftarrow f_0$
10: **end if**
11: **end for**return targetfunctions
12: **end procedure**



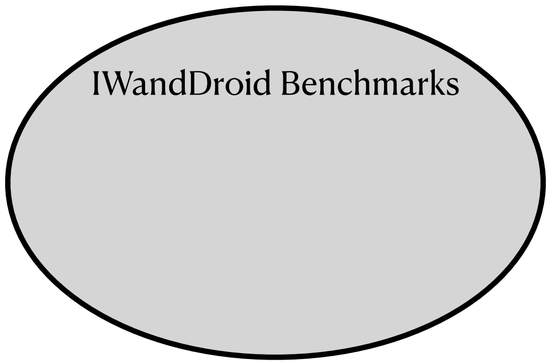
Line 2: Since the defined here, the edge is not propagated

Line 7: Triggers a backward analysis to find the target

Evaluation

Precision

- How does **IwanDroid** compare to the **HybriDroid** and **LuDroid**?



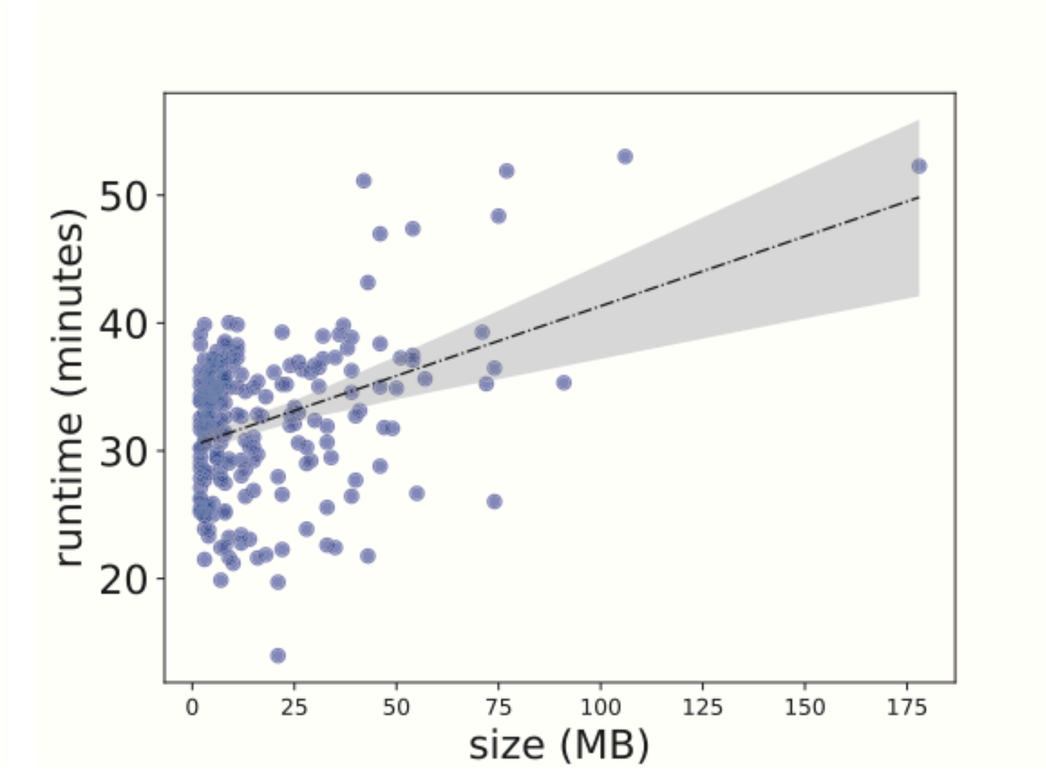
19 Microbenchmarks

App ID	App Name	JS Enabled	HybriDroid		LUDroid	IwanDroid
			version 2016	current version		
1	HelloCordova	no	NA	NA	NA	NA
2	HelloHybrid	yes	success	failed	success	success
3	HelloScript	yes	success	failed	failed	success
4	HelloScript_simple	yes	success	failed	success	success
5	HelloScript_test	yes	success	failed	success	success
6	HybridAPIArgNum	yes	success	failed	success	success
7	NormalAliasFlowTest	no	NA	NA	NA	NA
8	NormalAliasFlowTest_objfield_false	no	NA	NA	NA	NA
9	NormalAliasFlowTest_objfield1	no	NA	NA	NA	NA
10	StrongUpdate	yes	failed	failed	success	success
11	StrongUpdateCaseA	yes	failed	failed	success	success
12	StrongUpdateCaseB	yes	failed	failed	failed	success
13	StrongUpdateCaseC	yes	failed	failed	failed	success
14	JSUpdateCaseD	yes	failed	failed	failed	success
15	JSUpdateCaseE	yes	failed	failed	failed	success
16	JSUpdateCaseF	yes	failed	failed	failed	success
17	JSUpdateCaseG	yes	failed	failed	failed	success
18	DynamicJSCaseH	yes	failed	failed	failed	failed
19	DynamicAliasCaseI	yes	failed	failed	failed	failed

Evaluation

Scalability

- How scalable is IWanDroid on large application?



687 apps from F-Droid Store

Criteria:

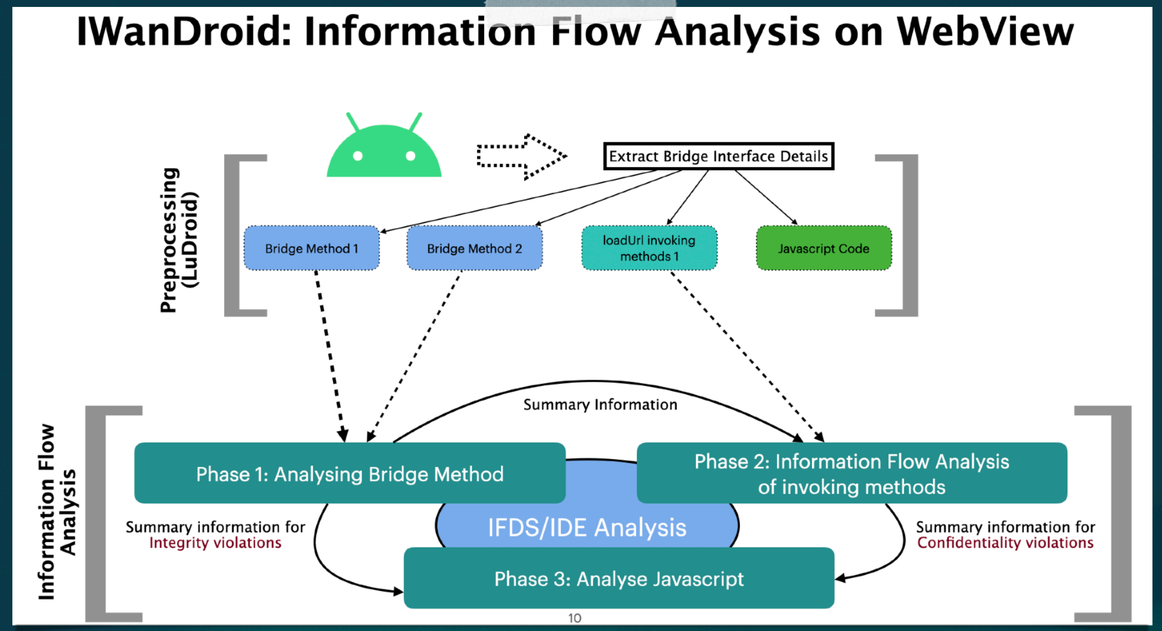
1. Relatively large apps
2. App should have `addJavaScript` enabled

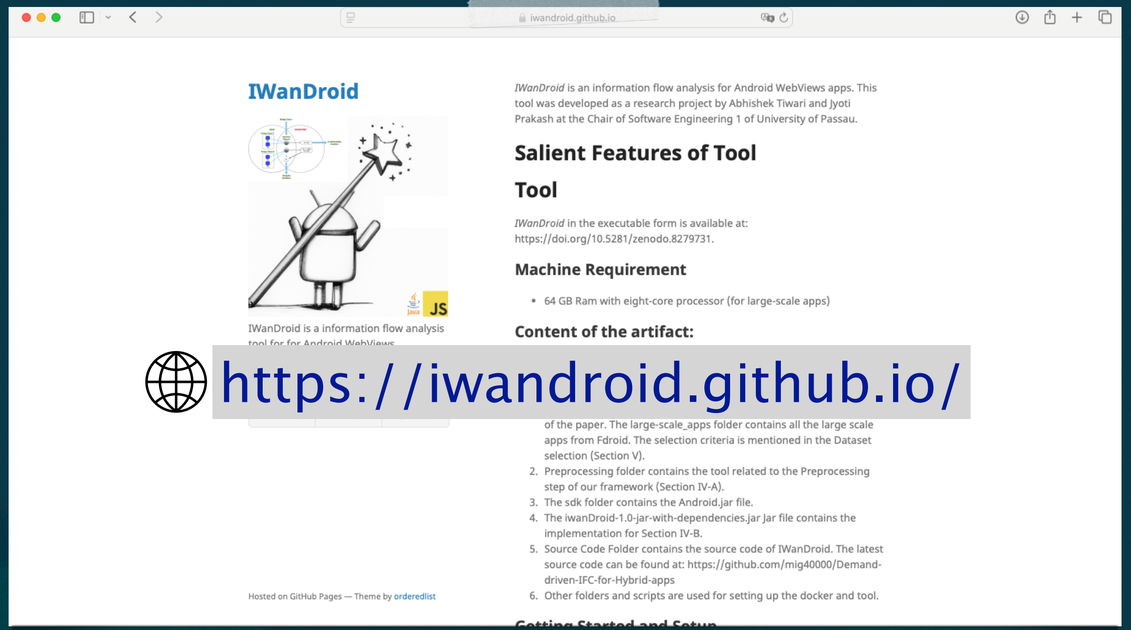
IwanDroid identified 136 confidentiality and 142 integrity violations.

Reflections

IWandDroid

- Defined a confidentiality and integrity threat model for Hybrid apps
- IWanDroid to detect the confidentiality and integrity violations without analysing the whole program





https://iwandroid.github.io/

Jyoti Prakash*
OpenText India

Abhishek Tiwari*
University of Southern Denmark

Christian Hammer
University of Passau