JSICA : 타이젠 앱 개인정보 누출 분석기 디자인 & 최적화

강동옥, 강지훈, 이광근
서울대학교 프로그래밍연구실
1/14/2014 in 紙之香

목표

- 타이젠 웹앱의 개인정보 누출을 탐지하는 분석기
  - 대상 : 타이젠 웹앱
    - 자바스크립트 + HTML5 + 타이젠 API + 라이브러리(jQuery)
  - 안전한(Sound) 분석
  - 합리적인 시간 안에 분석

결과

결과 : Tizen Sample App

	Code Size (LoC)	Time (s)	Mem (MB)	Alarm(#)	Leak Info.
analogWatch	96	1	23		
audioFilter	329	3	23		
bluetoothChat	1404	1	34	3	Bluetooth
callLog	1481	1	34		
chatter	1578	1	35	6	Phone Number
compass	84	1	23		
contactsExchanger	1394	1	34	42	Phone Number, NFC
deviceMotionCapture	86	1	23		
eventManager	1664	1	57		
exercisePlanner	1787	3	39		
fileManager	2030	1	35		
imageRotation	34	2	23		
mediaContent	262	1	34		
npRuntime	115	1	23		
offlineClockImage	105	2	23		
piano	130	1	23		
selfcamera	686	1	34	3	Photo
sensorBall	532	1	23		
systemInfo	179	1	34		
taskManager	224	6	23		
touchPaint	117	9	34		

결과 : Sunspider-1.0.2

	Code Size (LoC)	Time (s)	Mem (MB)
3d-cube	355	2	35
3d-morph	63	1	35
3d-raytrace	447	1	35
access-binary-trees	54	1	35
access-nbody	174	1	35
access-nsieve	46	1	20
bitops-3bit-bits-in-byte	40	1	20
bitops-bits-in-byte	30	1	20
bitops-bitwise-and	35	1	20
bitops-nsieve-bits	42	1	20
controlflow-recursive	32	1	20
crypto-md5	292	1	35
crypto-sha1	228	1	35
math-cordic	106	1	35
math-partial-sums	45	1	35
math-spectral-norm	60	1	35
regexp-dna	1721	1	42
string-base64	137	1	35
string-fasta	90	4	39

분석 디자인

감사합니다 SAFE

- 이미 표준 자바스크립트를 파싱할수 있는 SAFE 파서!

SAFE: Formal Specification and Implementation of a Scalable Analysis Framework for ECMAScript

Hongki Lee	Sooncheol Won	Joonho Jin	Junhee Cho	Sukyoung Ryu
KAIST	KAIST	KAIST	KAIST	KAIST
petitkan@kaist.ac.kr	wonsch@kaist.ac.kr	myfriend12@kaist.ac.kr	ssaljalu@kaist.ac.kr	sryu.cs@kaist.ac.kr

자바스크립트 핵심언어 디자인

Program	*pgm*	→	(*fid func*)* *fid*
Function	*func*	→	*f params decls* (*bid block*)* *bid*
Parameters	*params*	→	x^*
Declarations	*decls*	→	x^*
Block	*block*	→	*inst** *cont*
Instruction	*inst*	→	*lv* := *e*
		|	*x* := `func` *fid*
		|	*x* := `delete` *e e*
		|	`assert` *e*
		|	`addEvent` *e*
Continuation	*cont*	→	`jump` *bid**
		|	*x* := `call` *e e** ; `jump` *bid*
		|	*x* := `new` *e e** ; `jump` *bid*
		|	`return` *e*
		|	`invokeEvent` ; `jump` *bid*
Expression	*e*	→	*lv* | *c* | `this` | *e* ⊕ *e* | ⊖ *e*
LValue	*lv*	→	*x* | *e*[*e*]
Constant	*c*	→	*n* | ″*s*″ | `true` | `false` | `null` | `undef`
Variable	*x*		
FName	*f*		

챌린지

JavaScript = 클래스가 값 (Prototype)
+ 레코드 필드가 값
+ 함수가 값
+ 너무 많은 타입 변환 (Type Casting)
+ 많은 라이브러리 (jQuery, HTML DOM)
+ ~~Eval~~

레코드 필드가 값

```
x.concrete = 2;

x["concrete"] = 3;

y="rary";
x["arbit"+y] = "cd" ;
```

레코드 필드가 값

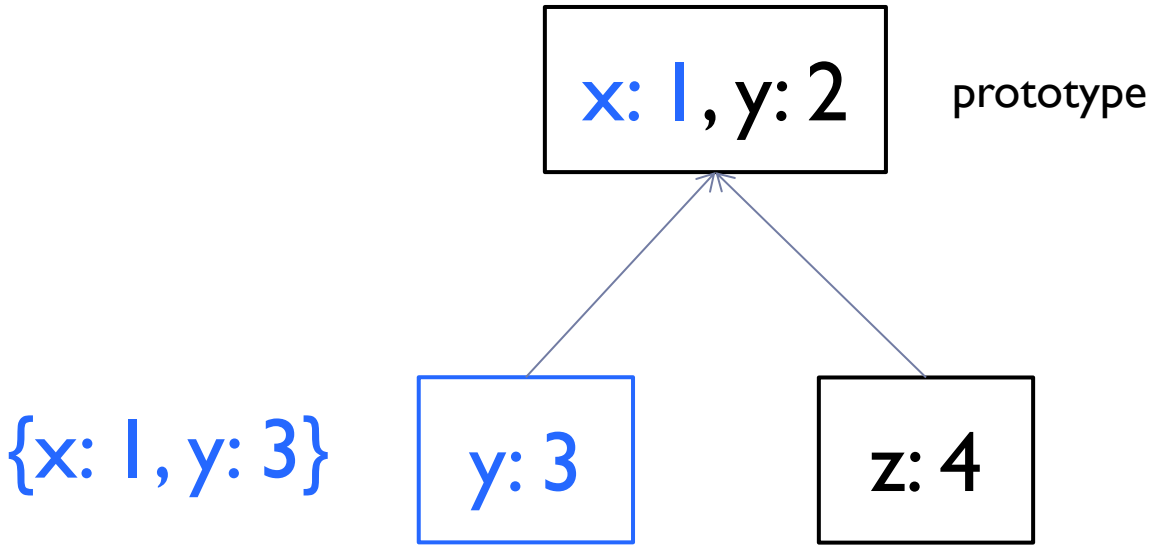
```
x.concrete = 2;

x["concrete"] = 3;

y="rary";
x["arbit"+y] = "cd" ;
```

x["a" + unknown()]
??

클래스가 값 (Prototype)



함수가 값

```
if ( unknown() ){
  func = function (){return 1}
}
else {
  func = function (){return 2}
}
func();
```

함수가 값 - Currying

```
function add(a,b){ return a+b;};
add2 = add.bind(2);
x=add2(4);
```

함수가 값 - Variable Capturing

```
function Person(){
  var a=1;
  this.grow = function(){ return ++a; }
}
(new Person()).grow() ;
```

요약 도메인

$$
\begin{aligned}
\hat{T} \in \hat{Trace} &= \Delta \xrightarrow{fin} \hat{State} \\
\hat{S} \in \hat{State} &= \hat{Heap} \times \hat{Store} \times \hat{This} \\
\hat{H} \in \hat{Heap} &= (\hat{Loc} \xrightarrow{fin} \hat{Obj}) \times (\hat{Loc} \xrightarrow{fin} 2^{\hat{Loc}}) \\
\hat{M} \in \hat{Store} &= (\hat{WeakVar} \xrightarrow{fin} \hat{Val}) \times (\hat{StrongVar} \xrightarrow{fin} \hat{Val}) \\
\hat{th} \in \hat{This} &= 2^{\hat{Loc}} \\
\hat{o} \in \hat{Obj} &= \hat{Closure} \times \hat{Record} \\
\hat{\rho} \in \hat{Record} &= \hat{Str} \xrightarrow{fin} (\hat{Val} \times \hat{Bool}) \\
\hat{cls} \in \hat{Closure} &= \hat{Fid} \xrightarrow{fin} \hat{Bindings} \\
\hat{Bindings} &= \hat{Value} \times \mathbb{N}_{\perp} \\
\hat{v} \in \hat{Val} &= 2^{\hat{Loc}} \times \hat{Bool} \times \hat{Num} \times \hat{Str} \times \{\text{undef}, \text{null}\}_{\perp} \\
\hat{u} \in \hat{Loc} &= AllocSite \\
\hat{s} \in \hat{Str} &= Str + Prefix
\end{aligned}
$$

Prototype 체인

레코드 필드

함수가 값

Currying

Variable Capturing

분석기 최적화

기본 분석 디자인의 문제

- 분석 속도 : 너무 느리다
  - Chatter 앱 (2kLoC) : 400s

분석 속도 최적화

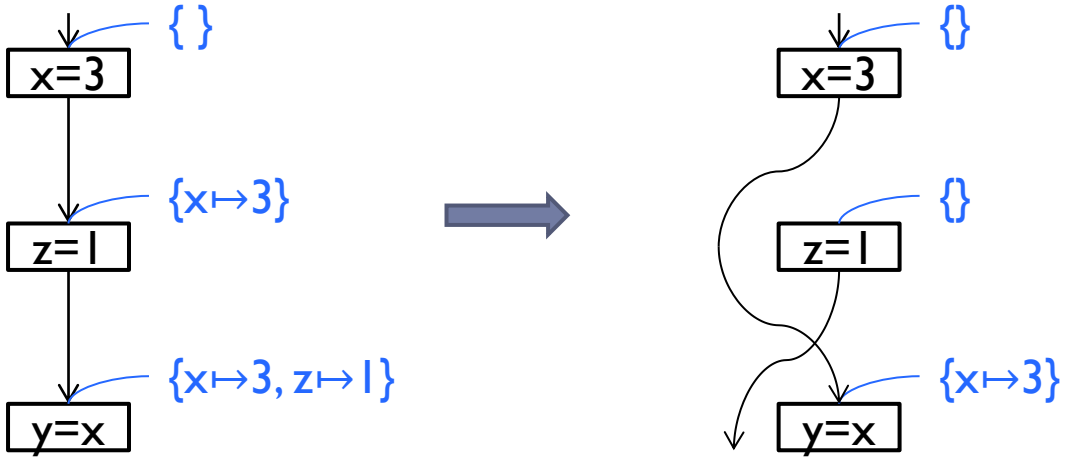
- 잘할 수 있는 것에 효율적으로 투자
  - Weak Update 하는 요약 메모리
    - 흐름 둔감 분석(Flow Insensitive)
  - Strong Update 하는 요약 메모리
    - 흐름 민감 분석(Flow Sensitive) + 스파스 분석(Sparse Analysis)

$$\hat{T} \in \hat{Trace} \quad = \quad \hat{Heap} \times \hat{WStore} \times (\Delta \xrightarrow{fin} \hat{Store} \times \hat{This})$$

Chatter (2KLoC) 400s ⇒ 1s

분석 속도 최적화

- Strong 변수는 Def/Use 정보를 미리 쉽게 구함
- 필요한 메모리만 필요한 곳으로



정리

정리

- 자유분방한 자바스크립트를 제대로 포섭
- 정확히 잘할 수 있는 것에 집중하고 효율적으로 분석
  - Heap, WStore는 흐름 둔감 분석
  - Store는 흐름 민감 분석 + 스파스 분석

앞으로 할일

- DOM 모델링
  - 자바스크립트는 각각의 HTML 태그를 오브젝트로 인식
    - 현재는 모든 DOM을 하나의 오브젝트로
  - 좀 더 정확한 분석을 위해 정교한 모델 필요
- Eval
  - 동적 생성 HTML 코드
    - 분석 중 자바스크립트 파싱 필요
  - 코드를 인자로 받을 수 있는 함수 ex) setTimeout

감사합니다

- Q&A

구현 이슈

- How?
  - State = Heap X Store
  - 하나는 flow insensitive
  - 하나는 flow sensitive

필요한 곳만 분석 (Temperal Locality)

- Heap, Wstore는 그냥 경로 둔감 분석만??
- 원하는 것 : 계산이 필요한 지점만 계산 하고 싶다
- 난점 : 필요한 지점을 미리 알기 어렵다
- 해결 : 필요한 지점을 모아가며 분석
  - 분석 중에
  - Location, 값 보존 변수(Weak Variable)의 Def,Use 정보를 모아서
  - 정의한 값(Def)이 쓰이(Use)는 지점만 워크리스트에

스파스 분석 구현

- 전역 요약 메모리 + SSA + Temperal Locality
  - 비보존 변수는 Alias가 없으므로 SSA 변환가능