FREEDOM: Engineering a State-of-the-Art DOM Fuzzer

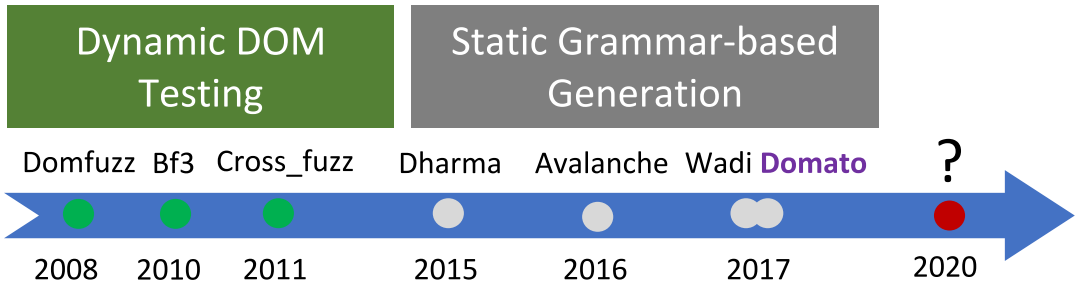
Wen Xu, Soyeon Park, and Taesoo Kim

Georgia Tech

ACM CCS 2020
November 9-13, 2020

DOM Engines have been studied for 10+ years

2009 - 2018



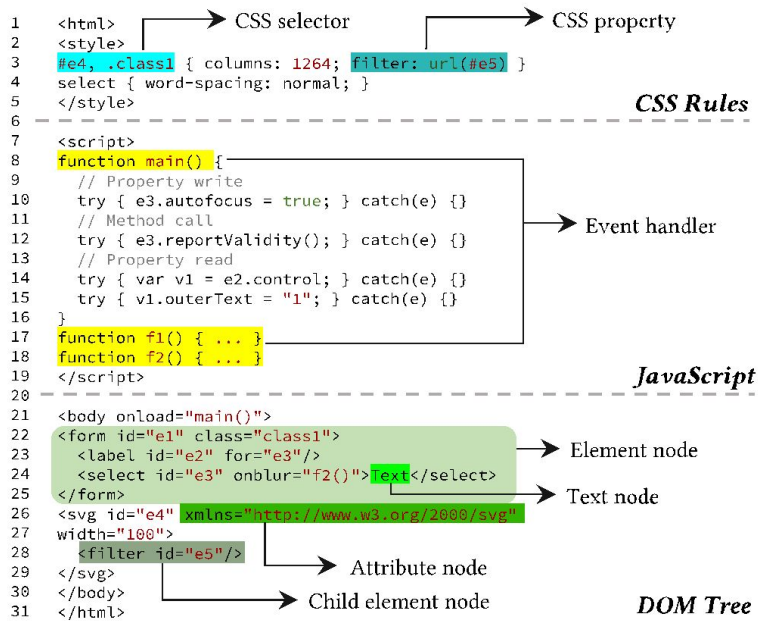
Do we really need a new DOM fuzzer?

× Fewer and fewer (exploitable) DOM bugs

× Heavily fuzzed mainstream DOM engines

✓ **FREEDOM:** A evolutionary DOM fuzzer finding new and more bugs!

Input Format: HTML Documents



CSS Rules specifies the styles of the DOM objects in the tree

JavaScript callbacks modifies the state (e.g., layout, effect, etc.) of the objects at runtime

DOM Tree specifies the objects to be displayed at the very beginning

Problems of the state of the art

Generating context-dependent values in HTML documents

✓ Certain types of values in a HTML document **refer to** other values

× Random generation based on context-free grammar cannot anticipate exact values concretized

<elementid> = htmlvar0000<int min=1 max=9>
<svgelementid> = svgvar0000<int min=1 max=9>
<class> = class<int min=0 max=9>
<tagname> = a | abbr | acronym | ...
<svgtagname> = a | altGlyph | altGlyphDef | ...

<selector> = .<class>
<selector> = #<elementid>
<selector> = <element>
<element p=0.5> = <tagname>
<element p=0.4> = <svgtagname>

A random document probably has **30** HTML elements, only **5** SVG elements or simply **no <a>**

<svgelement_animate> = <lt>animate <animattr> <svgattrs_animate> /<gt>
<animattr> = attributeName="x" from="<x_value>"
<animattr> = attributeName="y" from="<y_value>"
<animattr> = attributeName="d" from="<d_value>"
... and many more.

A random document probably mutates the non-existent **x** attribute of **<path>**

Check the paper for four types of context dependences that existing DOM fuzzers are unable to describe

Mutation-based DOM Fuzzing

× Whether or not coverage-guided mutation works against DOM engines is an open problem

× Stateless HTML files in plaintext are not fully mutable

Our solution: FREEDOM
A modern DOM fuzzing playground

* Grammar rules selected from Domato

FREEDOM Can Still Find New Critical DOM Bugs

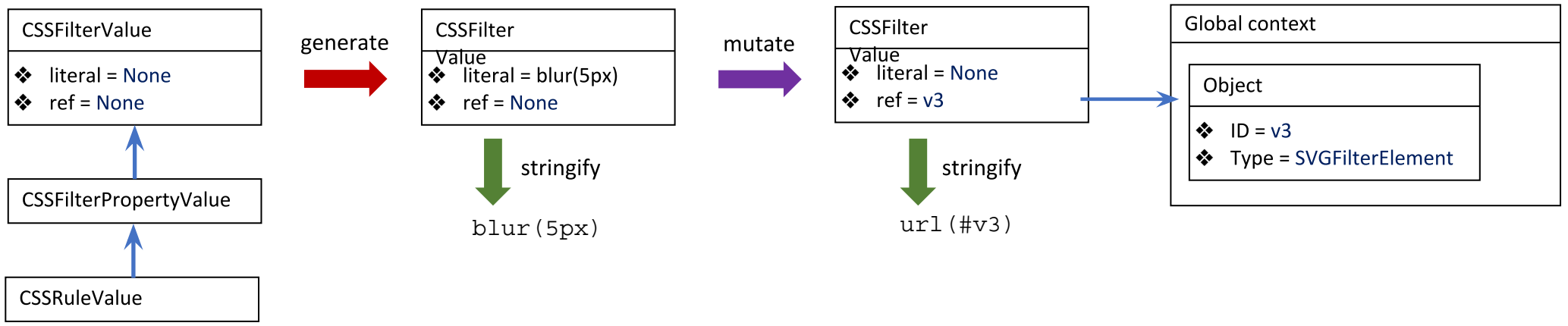
24 0days in Safari, Chrome, and Firefox during 2019 and 2020

10 CVEs and 65K USD bug bounty

A macOS remote exploit based on one 0day found in Safari

FD-IR: A stateful HTML intermediate representation

Two types of contexts and numerous interdependent, lowerable, generative and mutable **Value** instances



Standard Compliance

- ✓ Covers DOM, CSS, and JavaScript
- ✓ Lowered to HTML files in plaintext
- ✓ Highly extensible

Context-Awareness

- ✓ Global object and token pools
- ✓ Function-wise object pools (U-D)
- ✓ References between DOM objects

Fuzzing Support

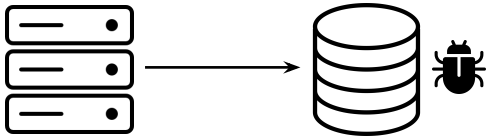
- ✓ Random generation from scratch
- ✓ Mutation over existing FD-IR
- ✓ Merging two FD-IR programs

A Python3 implementation with serialization and deserialization support

FREEDOM Workflow

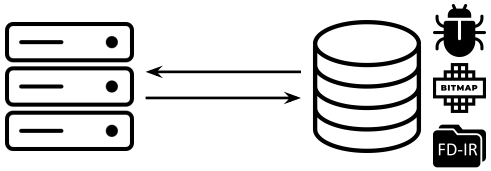
A **cloud-based** setting with **two** working modes

Generation-based Blackbox Fuzzing



- Generate a FD-IR program through **6** appending algorithms
- Lower to HTML files for testing
- Report crash to the central server

Mutation-based Coverage-guided Fuzzing



- Randomly mutate a FD-IR program fetched from the server in 16 ways
- Profile runtime coverage
- Upload testcases visiting new code blocks in FD-IR to server database

Comparison With State-of-the-Arts

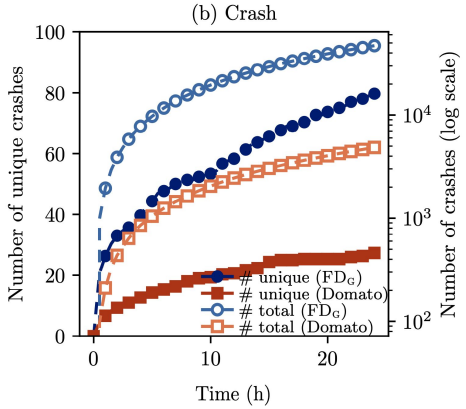
Fuzzing WebKit generatively with 100 cores for 24 hours

Dharma

✓ Visits 13.96% more code blocks

✓ 7 unique crashes on average (Dharma found none)

Domato



✓ Similar code coverage

✓ 3x more unique crashes

✓ 3x more security-related crashes

✓ Covers Nearly 90% Domato crashes

✓ Nearly 65% crashes have context-dependent values that Domato hardly generates

FREEDOM becomes the new state-of-the-art

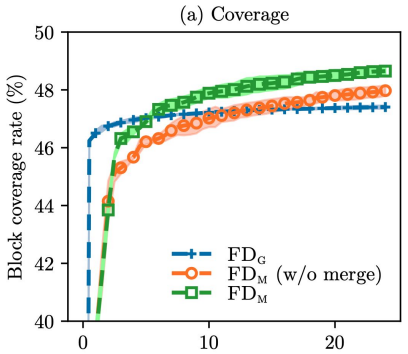
✓ Captures the old types of bugs described by context-free grammars

✓ Discovers new bugs that require semantically complicated inputs

Exploring Coverage-Driven DOM Fuzzing

Fuzzing WebKit mutationally driven by coverage (blank corpus) and compare with blackbox generation

Effectiveness



✓ 2.62% more coverage
✓ 3 new unique crashes
✓ 2 security-related crashes
✓ More likely to trigger crashes that require strict or subtle settings

Ineffectiveness

× Nearly 3.8x fewer unique crashes on average
× Misses around 75% crashes by generation
× Misses 7 security-related crashes

Summary

- Blackbox generation is not comprehensive but efficient and effective in a reasonable time
- Coverage-guided mutation is irreplaceable for finding hard-to-reach bugs
- Improving coverage-guided DOM fuzzing with more resources and existing corpus

Check the paper for concrete bug cases that are found by coverage-driven fuzzing but missed by random generation

* FD$_G$: FREEDOM in generation mode
FD$_M$: FREEDOM in coverage-guided mutation mode

Conclusion

- FREEDOM: An evolutionary static DOM fuzzer
  - Supporting both generation-based and coverage-guided mutation-based fuzzing
  - Based on a context-aware IR for HTML documents
  - Finding new DOM engine bugs
- First study on applying coverage-guided mutation to DOM Fuzzing
- We expect further improvement on DOM fuzzing facilitated by FREEDOM
- Open sourced at https://github.com/sslab-gatech/freedom