EVENTRACER: CONCURRENCY ANALYSIS FOR EVENT-DRIVEN APPLICATIONS

Martin Vechev

ETH Zurich, Software Reliablity Lab

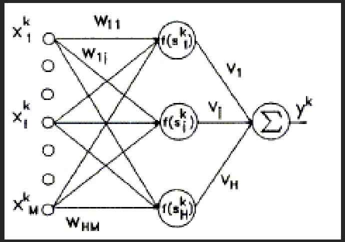
http://www.srl.inf.ethz.ch

Research @SRL (Sample)

EVENT RACER

Analysis of Event-Driven Applications

http://eventracer.org



"Big Code" Analytics

e.g. http://jsnice.org

Fender: Programming with Relaxed Models

http://practicalsynthesis.org/fender/

PLDI’12, OOPSLA’13, PLDI’14, StrangeLoop’14

PLDI’14, Onward’14, JSNice

FMCAD’10, PLDI’11, PLDI’12 SAS’13, SAS’14

more info: http://www.srl.inf.ethz.ch/

EventRacer

EVENT RACER

Analysis of Event-Driven Applications

http://eventracer.org

PLDI'12, OOPSLA'13, PLDI'14, StrangeLoop'14

People:

- ETH: Martin Vechev, Veselin Raychev, Pavol Bielik, Jeremie Miserez
- Princeton: Laurent Vanbever
- Aarhus: Anders Moeller, Casper Jensen
- Samsung Research: Manu Sridharan
- Sofia University: Boris Petrov, Yasen Trifonov
- IBM T.J Watson: Julian Dolby

Research spanning runtime systems, program analysis, algorithms and theory

Event-Driven: Motivation

EDITION: INTERNATIONAL | U.S. | MÉXICO | ARABIC
TV: CNNi | CNN en Español
Set edition preference
Home Video World U.S. Africa Asia Europe Latin America Middle East Business World S
EDITOR'S CHOICE Scenes from the field • Pistorius verdict • ISIS threat • Apple Watch • Scotland vote

September 11, 2014 -- Updated 0936 GMT (1736 HKT) Edited by Peter Wilkinson in London

Obama: 'We will degrade, ultimately destroy' ISIS

"Difficult speech" for Barack Obama: President announces U.S. airstrikes will hit targets in war-ravaged Syria as well as Iraq. FULL STORY

Pistorius not guilty of murder: Judge

Oscar Pistorius cannot be found guilty of murdering his girlfriend, the judge rules. The Olympian faces other charges, such as culpable homicide, and awaits a formal verdict. STORY | LIVE UPDATES

~ 1 trillion websites today

~ 1 billion smartphones

Reacts to events: user clicks, arrival of network requests

Event-Driven: Motivation

Obama: 'We will degrade, ultimately destroy' ISIS

"Difficult speech" for Barack Obama: President announces U.S. airstrikes will hit targets in war-ravaged Syria as well as Iraq. FULL STORY

Pistorius not guilty of murder: Judge

Oscar Pistorius cannot be found guilty of murdering his girlfriend, the judge rules. The Olympian faces other charges, such as culpable homicide, and awaits a formal verdict. STORY | LIVE UPDATES

~ 1 trillion websites today

Wanted: fast response time

~ 1 billion smartphones

Reacts to events: user clicks, arrival of network requests

Event-Driven: Motivation

CNN
EDITION: INTERNATIONAL | U.S. | MÉXICO | ARABIC
TV: CNNi | CNN en Español
Set edition preference
Home Video World U.S. Africa Asia Europe Latin America Middle East Business World S
EDITOR'S CHOICE
Scenes from the field · Pistorius verdict · ISIS threat · Apple Watch · Scotland vote

Obama: 'We will degrade, ultimately destroy' ISIS

"Difficult speech" for Barack Obama: President announces U.S. airstrikes will hit targets in war-ravaged Syria as well as Iraq. FULL STORY

Pistorius not guilty of murder: Judge

Oscar Pistorius cannot be found guilty of murdering his girlfriend, the judge rules. The Olympian faces other charges, such as culpable homicide, and awaits a formal verdict. STORY | LIVE UPDATES

~ 1 trillion websites today

Wanted: fast response time

~ 1 billion smartphones

Reacts to events: user clicks, arrival of network requests

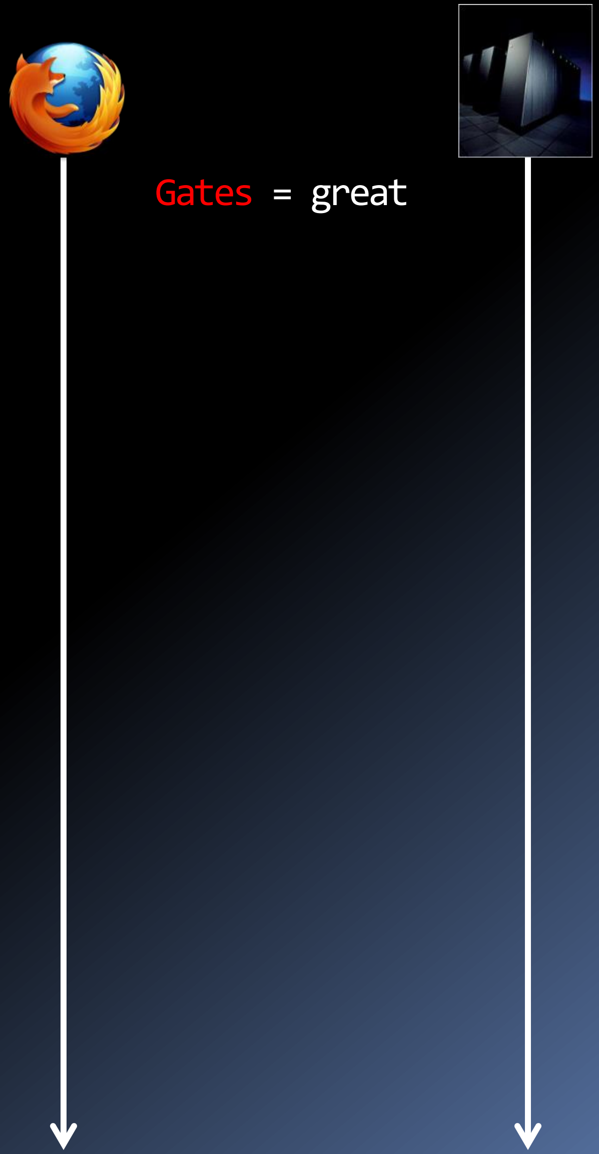
Highly Asynchronous, Complex control flow

Non-determinism: network latency

```
<html>
<head></head>

<body>
<script>
var Gates = “great”;</script>
</script>

<img src=“img1.png” onload=“Gates=‘poor’;”>
<img src=“img2.png” onload=“alert(Gates);”>

</body>
</html>
```

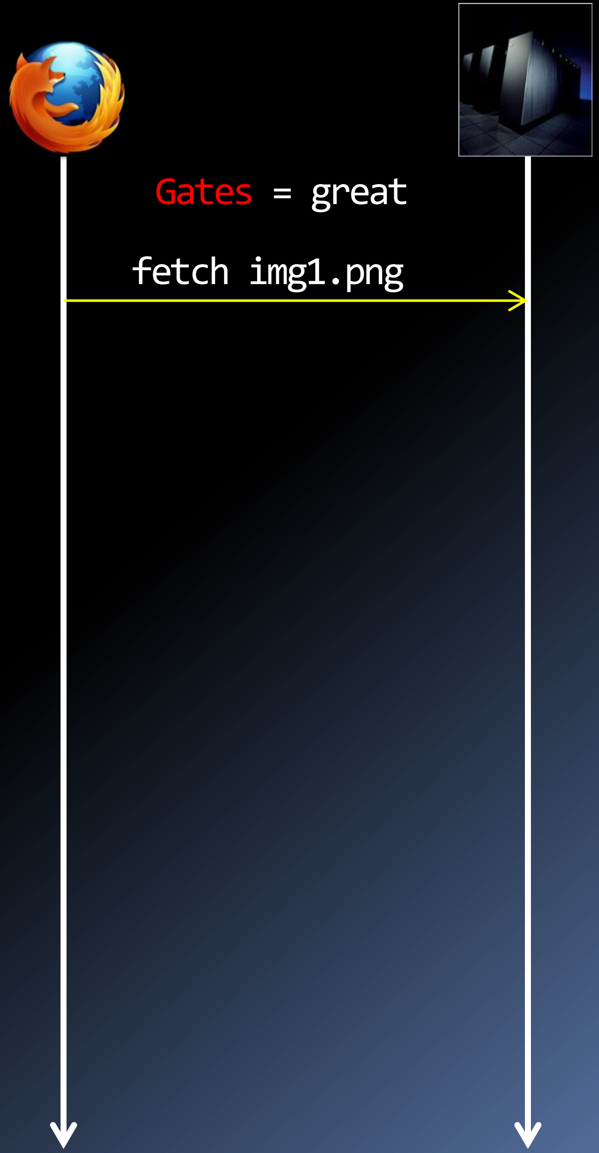
Non-determinism: network latency

```
<html>
<head></head>

<body>
<script>
var Gates = “great”;</script>
</script>

<img src=“img1.png” onload=“Gates=‘poor’;”>
<img src=“img2.png” onload=“alert(Gates);”>

</body>
</html>
```



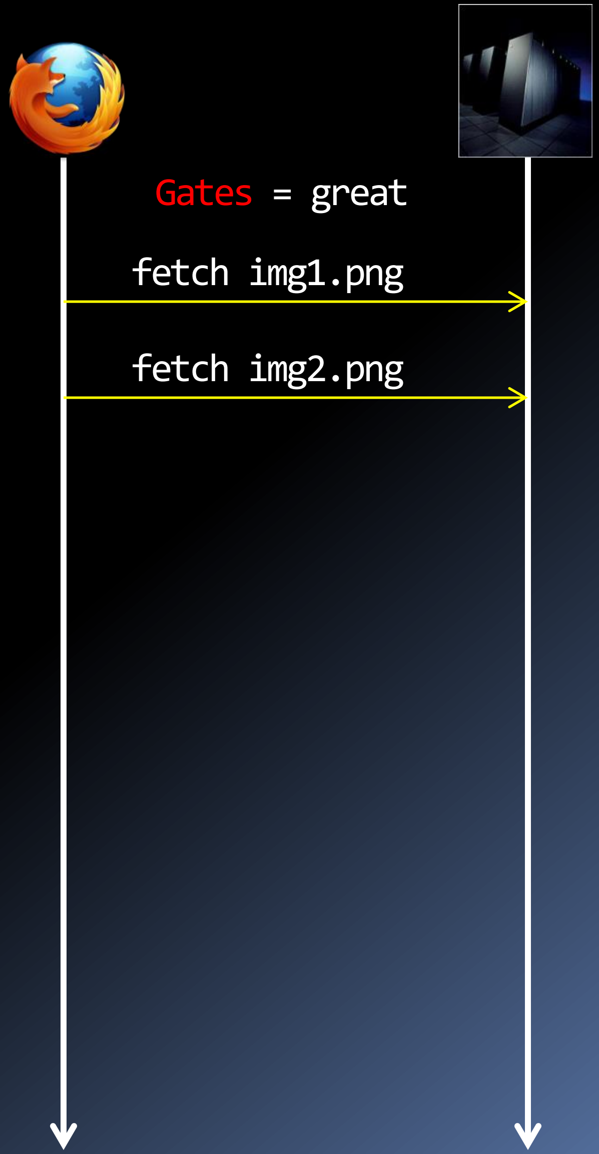
Non-determinism: network latency

```
<html>
<head></head>

<body>
<script>
var Gates = “great”;</script>
</script>

<img src=“img1.png” onload=“Gates=‘poor’;”>
<img src=“img2.png” onload=“alert(Gates);”>

</body>
</html>
```



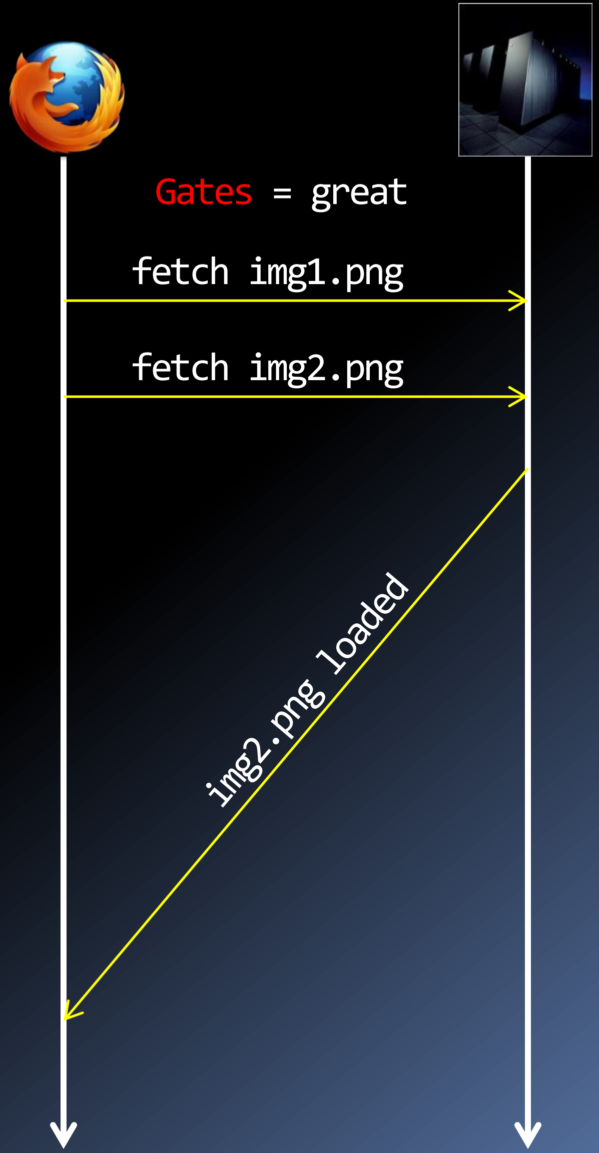
Non-determinism: network latency

```
<html>
<head></head>

<body>
<script>
var Gates = “great”;</script>
</script>

<img src=“img1.png” onload=“Gates=‘poor’;”>
<img src=“img2.png” onload=“alert(Gates);”>

</body>
</html>
```



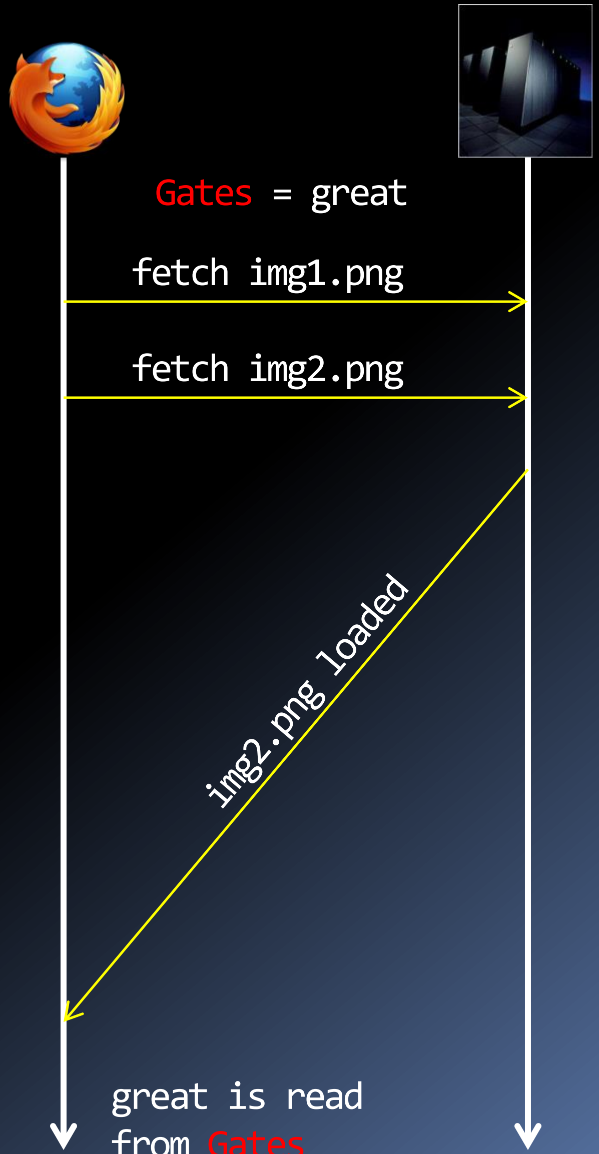
Non-determinism: network latency

```
<html>
<head></head>

<body>
<script>
var Gates = “great”;</script>
</script>

<img src=“img1.png” onload=“Gates=‘poor’;”>
<img src=“img2.png” onload=“alert(Gates);”>

</body>
</html>
```



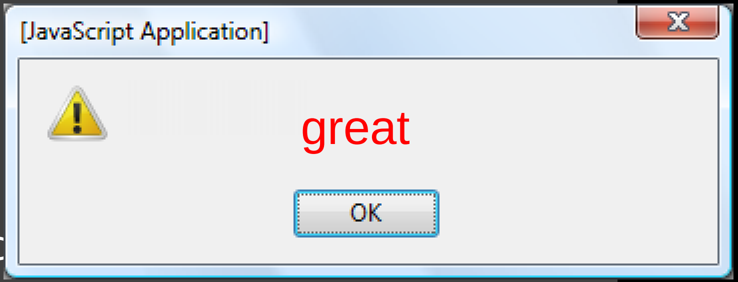
Non-determinism: network latency

```
<html>
<head></head>

<body>
<script>
var Gates = "great";</script>
</script>

<img src="img1.png" onload="Gates='poor';">
<img src="img2.png" onload="alert(Gates);">

</body>
</html>
```

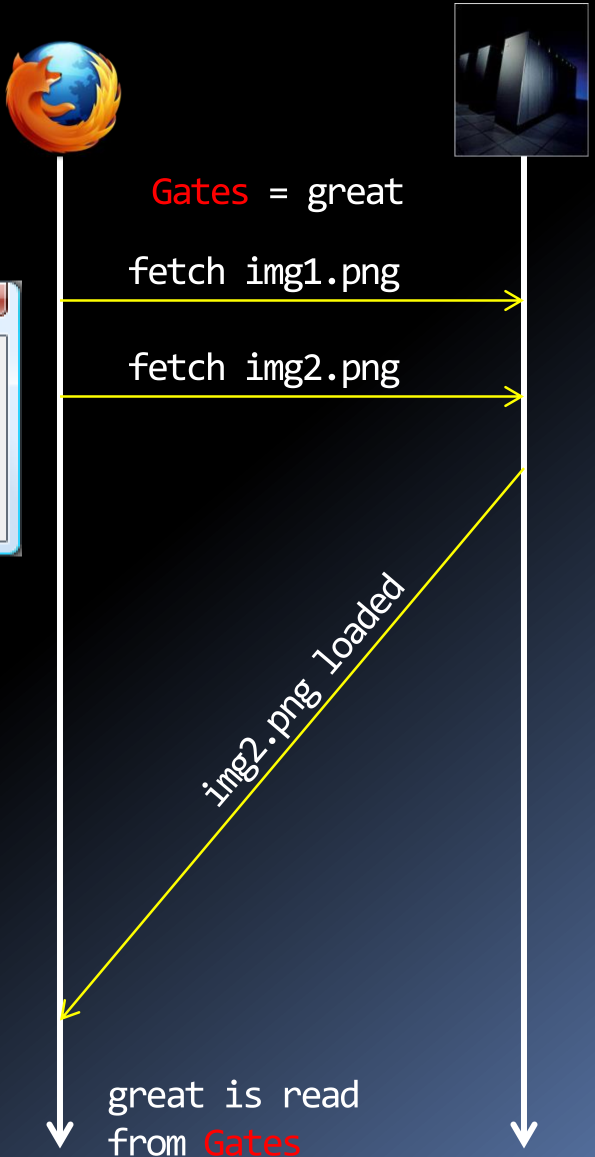


Non-determinism: network latency

```
<html>
<head></head>

<body>
<script>
var Gates = "great";</sc
</script>

<img src="img1.png" onload="Gates='poor';">
<img src="img2.png" onload="alert(Gates);">

</body>
</html>
```

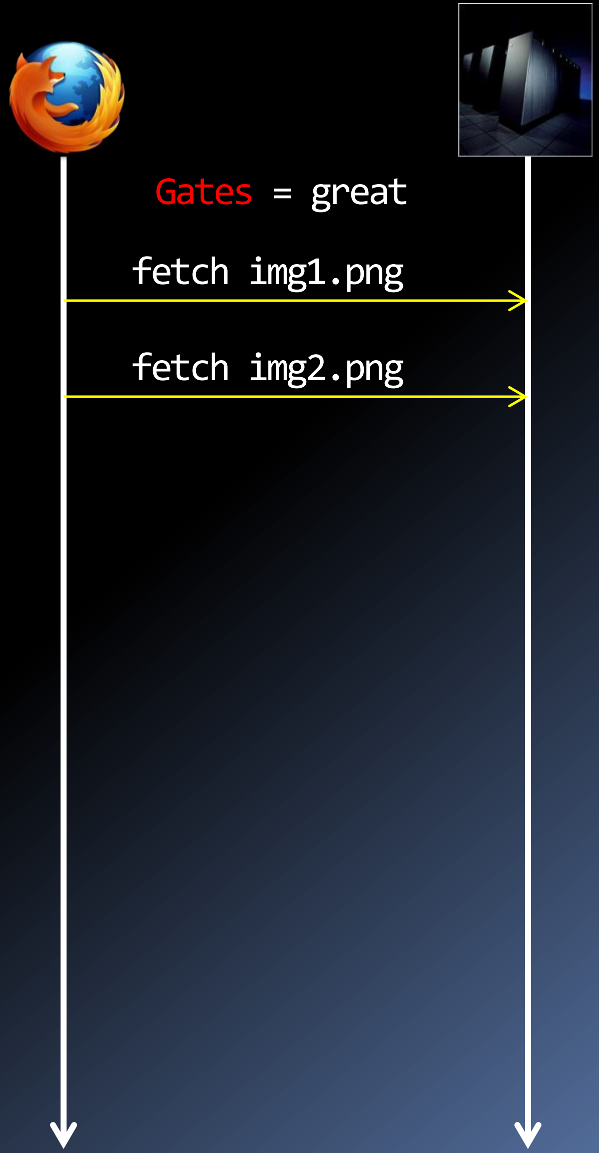

[JavaScript Application]

great



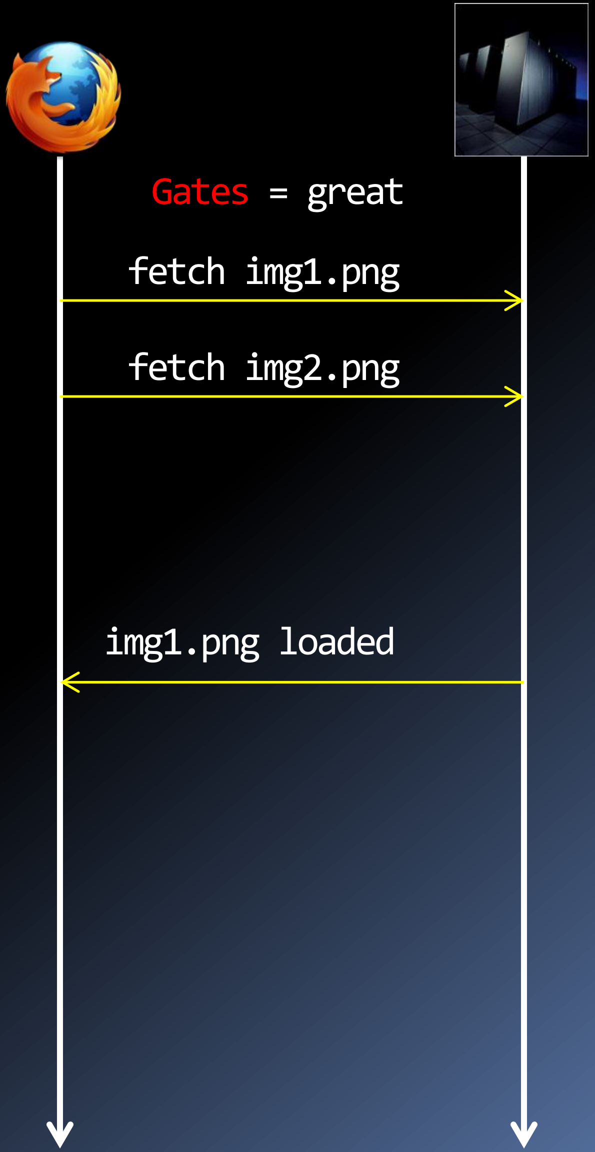
Non-determinism: network latency

```
<html>
<head></head>

<body>
<script>
var Gates = "great";</script>
</script>

<img src="img1.png" onload="Gates='poor';">
<img src="img2.png" onload="alert(Gates);">

</body>
</html>
```



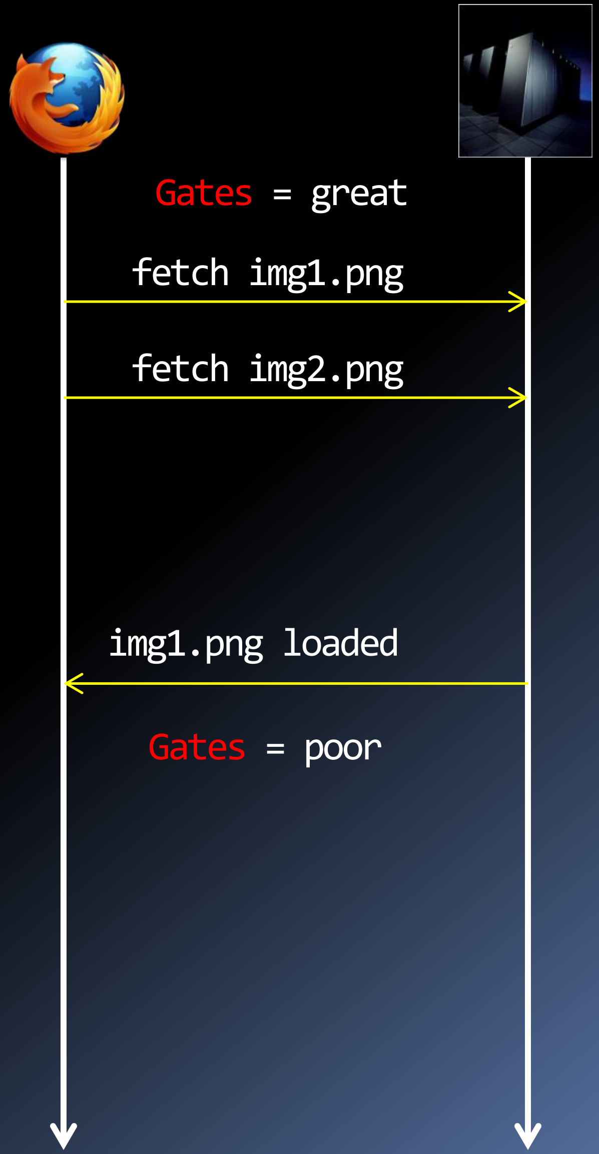
Non-determinism: network latency

```
<html>
<head></head>

<body>
<script>
var Gates = “great”;</script>
</script>

<img src=“img1.png” onload=“Gates=‘poor’;”>
<img src=“img2.png” onload=“alert(Gates);”>

</body>
</html>
```



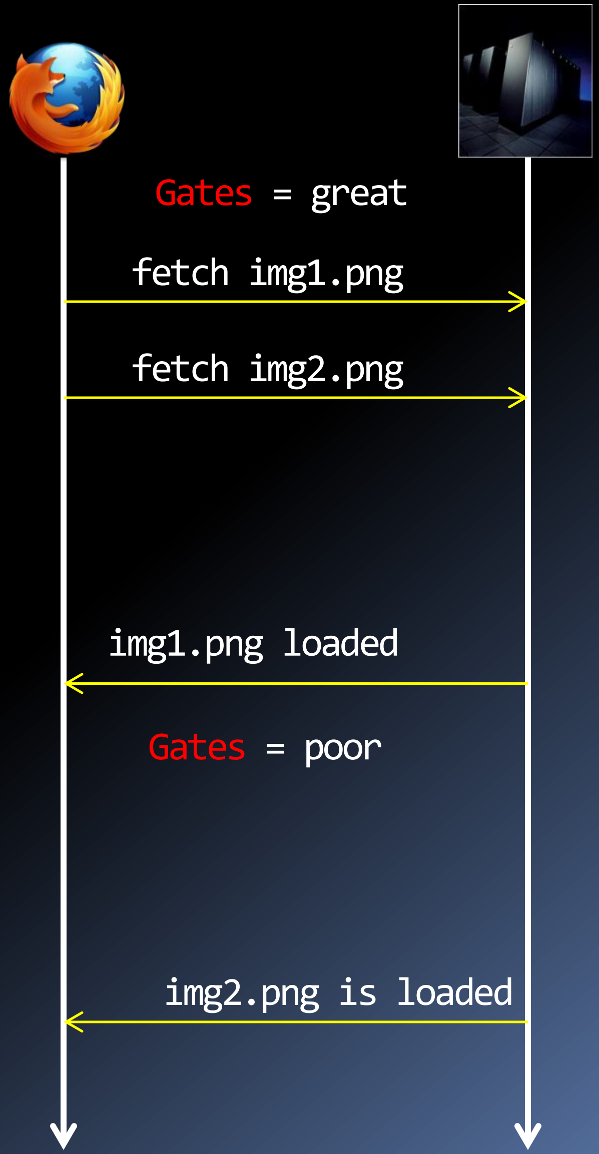
Non-determinism: network latency

```
<html>
<head></head>

<body>
<script>
var Gates = “great”;</script>
</script>

<img src=“img1.png” onload=“Gates=‘poor’;”>
<img src=“img2.png” onload=“alert(Gates);”>

</body>
</html>
```



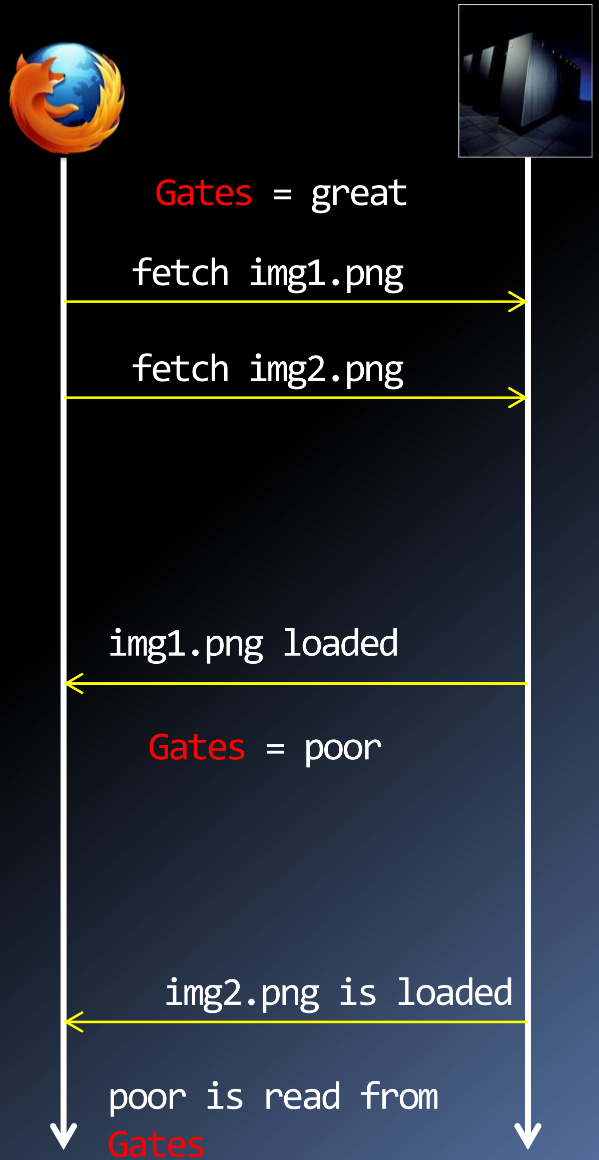
Non-determinism: network latency

```
<html>
<head></head>

<body>
<script>
var Gates = “great”;</script>
</script>

<img src=“img1.png” onload=“Gates=‘poor’;”>
<img src=“img2.png” onload=“alert(Gates);”>

</body>
</html>
```



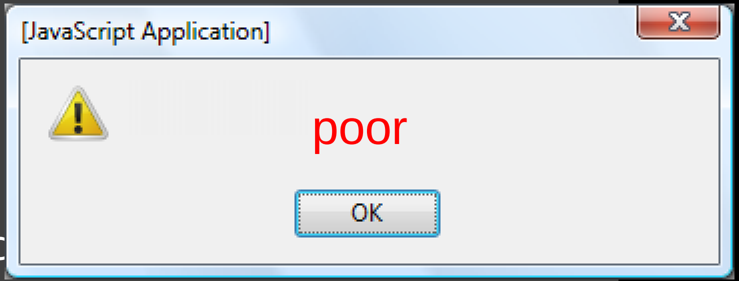
Non-determinism: network latency

```
<html>
<head></head>

<body>
<script>
var Gates = "great";</script>
</script>

<img src="img1.png" onload="Gates='poor';">
<img src="img2.png" onload="alert(Gates);">

</body>
</html>
```

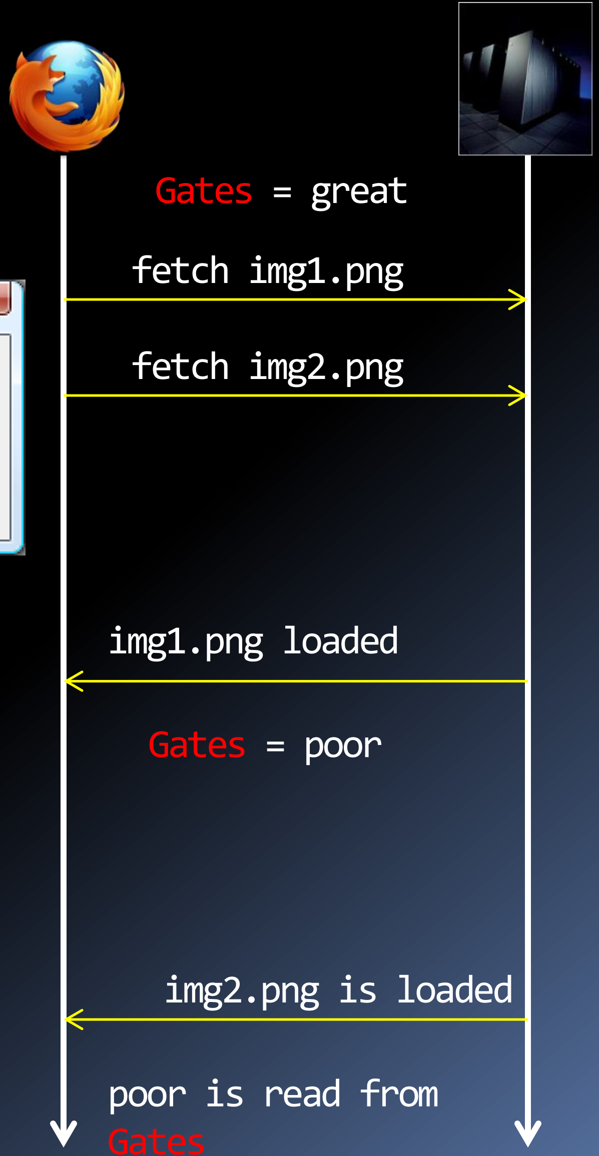


Non-determinism: network latency

```
<html>
<head></head>

<body>
<script>
var Gates = “great”;</sc
</script>

<img src=“img1.png” onload=“Gates=‘poor’;”>
<img src=“img2.png” onload=“alert(Gates);”>

</body>
</html>
```



poor

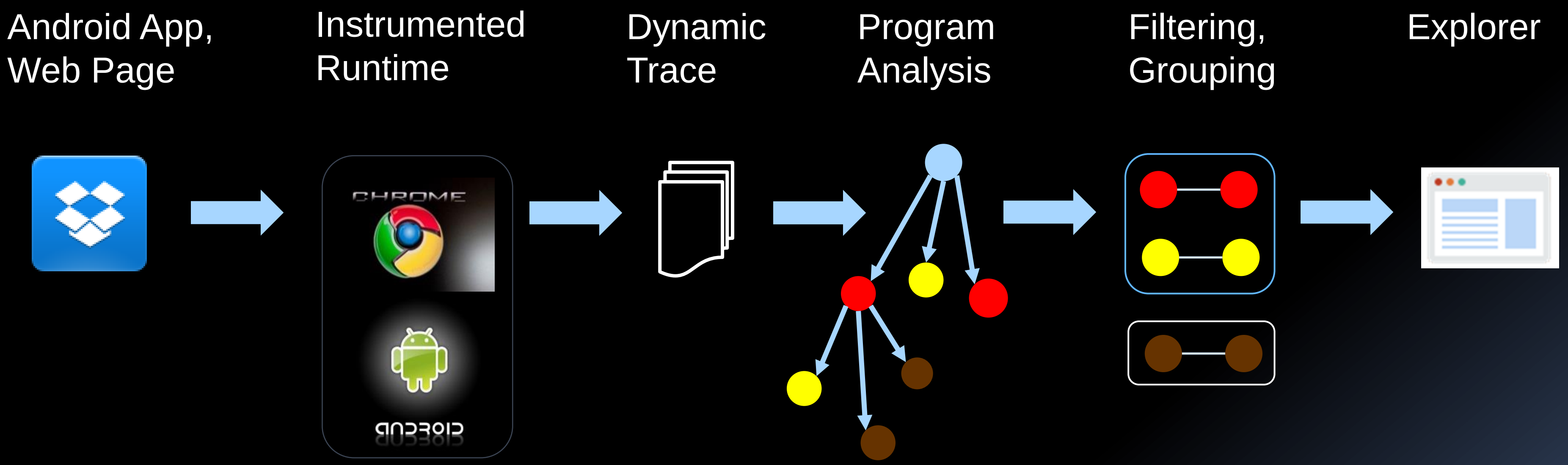


What do we learn from these?

- Asynchrony causes non-determinism which may cause unwanted behavior
- Non-determinism is caused by interfering unordered accesses to shared locations

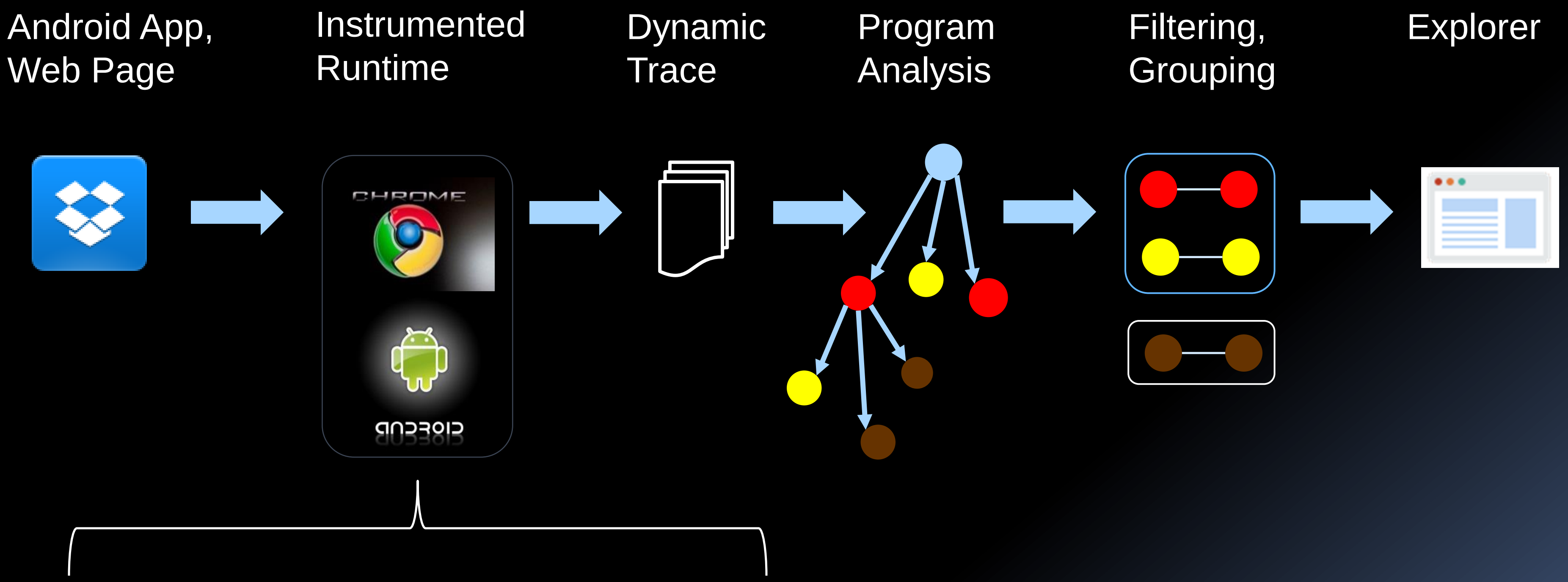
Can we build a system that detects such violations?

EventRacer Flow



Demo

EventRacer Flow



- <span>Memory locations:</span> JavaScript vars, functions, HTML DOM elements, etc

- Happens-before events: atomic actions (e.g. parsing HTML element), events that order actions

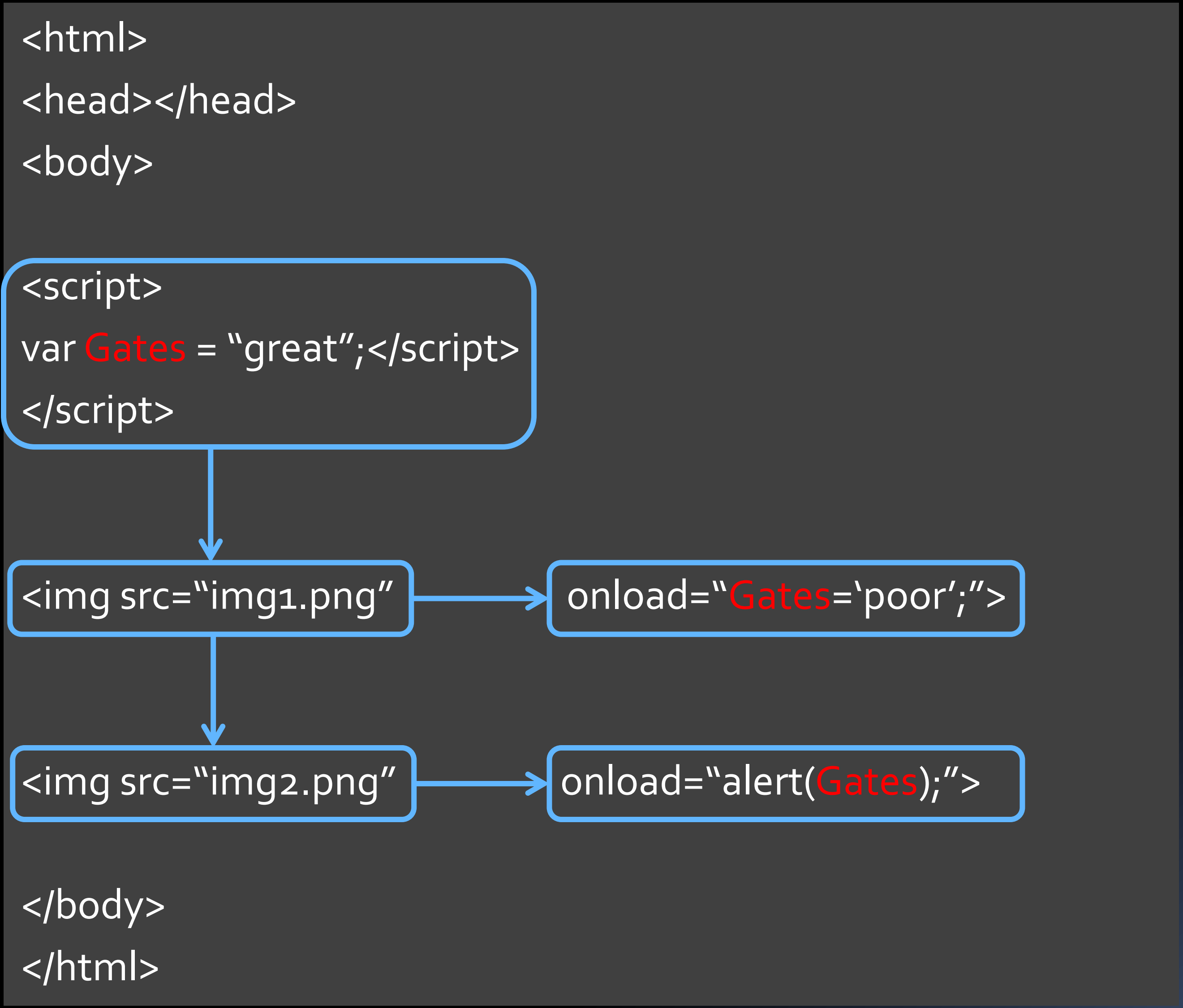
Example of Happens-Before

```
<html>
<head></head>
<body>

<script>
var Gates = “great”;</script>
</script>



<img src=“img1.png”            onload=“Gates=‘poor’;”>



<img src=“img2.png”            onload=“alert(Gates);”>


</body>
</html>
```

Example of Happens-Before

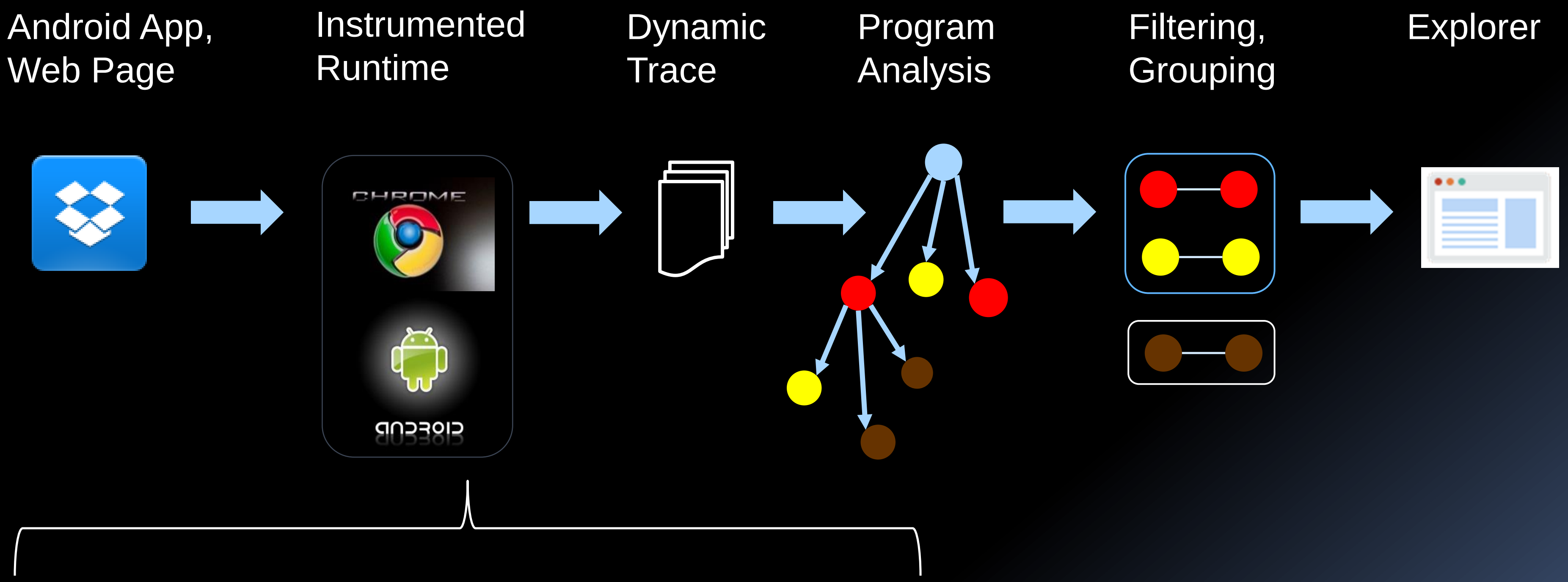
```
<html>
<head></head>
<body>

<script>
var Gates = “great”;</script>
</script>


<img src=“img1.png”          onload=“Gates=‘poor’;”>


<img src=“img2.png”          onload=“alert(Gates);”>

</body>
</html>
```

Example of Happens-Before



```
<html>
<head></head>
<body>

<script>
var Gates = “great”;</script>
</script>

<img src=“img1.png”    onload=“Gates=‘poor’;”>

<img src=“img2.png”    onload=“alert(Gates);”>

</body>
</html>
```

EventRacer Flow



- Works with V8 and Chrome: AST re-writes inside V8. Very efficient, reuses some analysis (e.g. local vs. global)

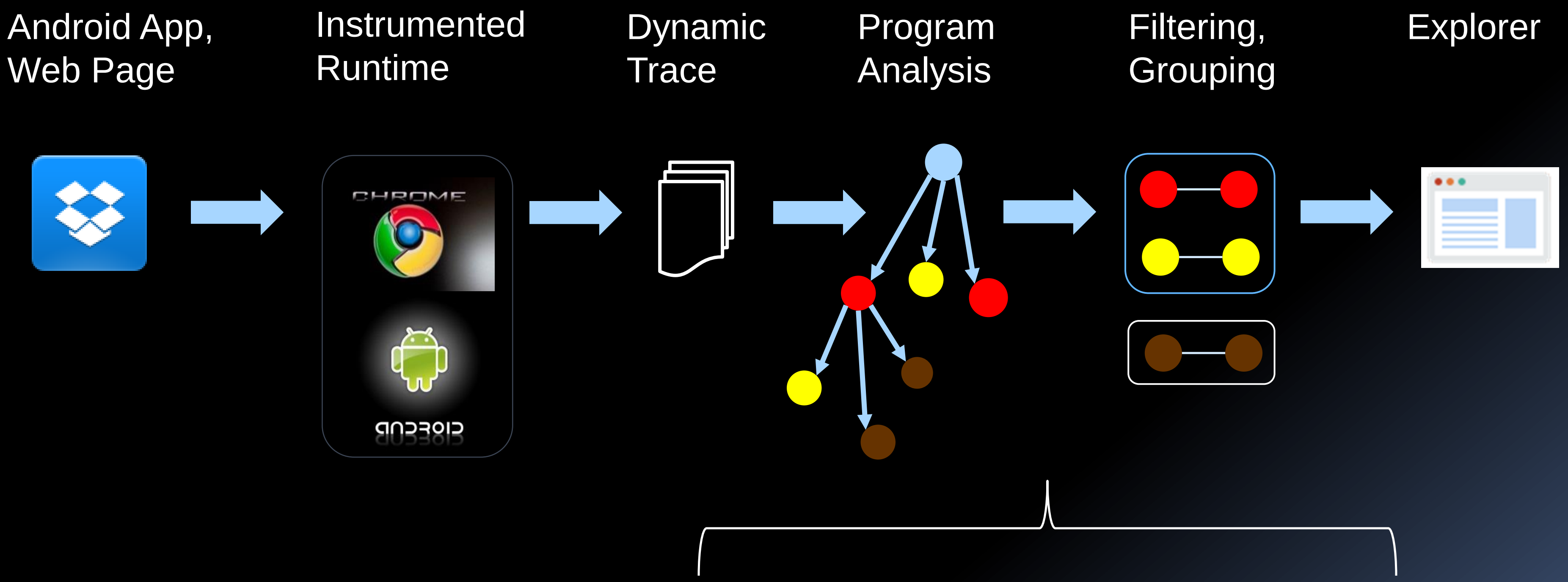
- Care is to be taken due to multiple ASTs !

Program Analysis: Two Key Challenges

Precision: too many false positives caused by synchronization with read/writes

Scalability: due to too many event handlers overwhelming the analysis data structures

EventRacer Flow



- New graph algorithms with good space/time complexity: combine chains + vector clocks, ~100x space reduction
- Notions of coverage + filters/grouping: ~30x reduction on reported false positives

Evaluation – Android

- Real-world apps, e.g.
- Takes 20-30 seconds to analyze 10 min long interactions
  - cannot be analyzed by any existing system
- Reports few violations, many harmful ones
  - Some reported bugs fixed by developers

Evaluation – Web Pages

- On Fortune 100 web sites
- Takes seconds to analyze complex interactions
- Reports ~17 violations per web page
  - 25% harmful, 57% synchronization, 17% harmless

EventRacer Flow

Android App, Web Page

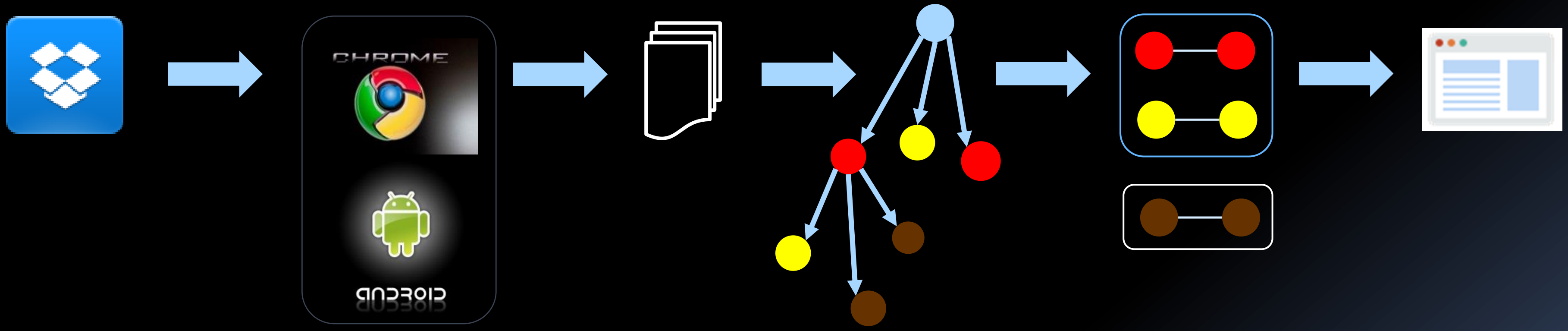
Instrumented Runtime

Dynamic Trace

Program Analysis

Filtering, Grouping

Explorer



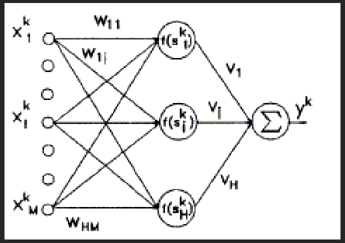
http://eventracer.org

Research @SRL (Sample)

EVENT RACER

Analysis of Event-Driven Applications

http://eventracer.org



"Big Code" Analytics

e.g. http://jsnice.org

Fender: Programming with Relaxed Models

http://practicalsynthesis.org/fender/

More info: http://www.srl.inf.ethz.ch/