Automated Black-Box Detection of Side-Channel Vulnerabilities in Web Applications

Peter Chapman
David Evans
University of Virginia
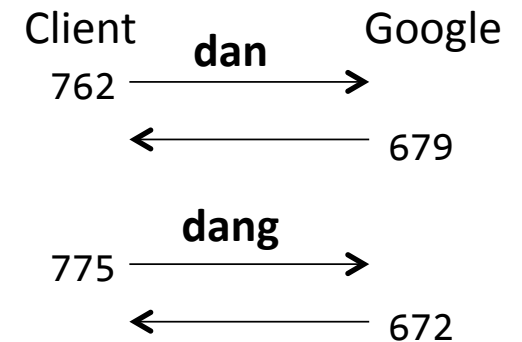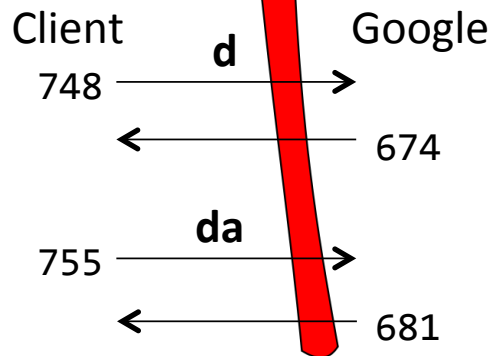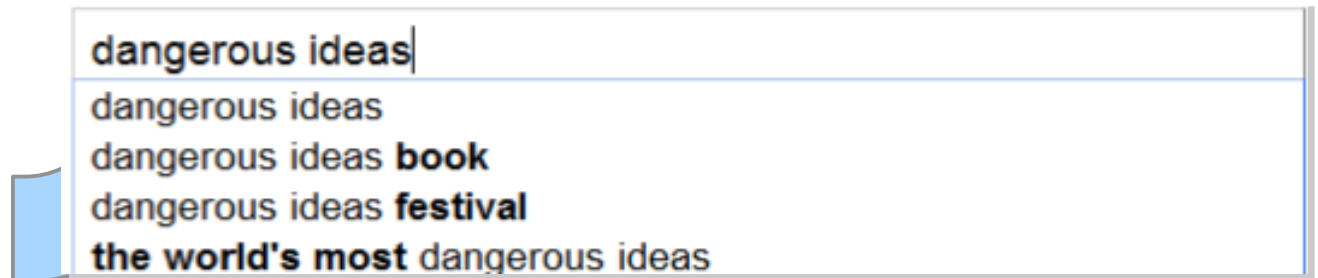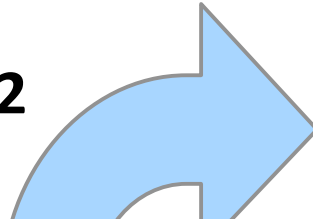http://www.cs.virginia.edu/sca/

CCS '11
October 19, 2011

Side-Channel Leaks in Web Apps

HTTPS over WPA2


```
dangerous ideas|
dangerous ideas
dangerous ideas book
dangerous ideas festival
the world's most dangerous ideas
```

Client	**d**	Google
748	→	
	←	674
755	**da** →	
	←	681

Client	**dan**	Google
762	→	
	←	679
775	**dang** →	
	←	672

Chen[+], Oakland 2010

Modern Web Apps

Dynamic and Responsive Browsing Experience

On-Demand Content

HTML 5

↓ Traffic

↓ Latency

↑ Responsiveness

Traffic is now closely associated with the demanded content.

Motivation: Detect Vulnerabilities

Motivation: Evaluate Defenses

Randomized or Uniform Communication Attributes

Packet Sizes

Transfer Control Flow

Inter-Packet Timings

Requests and Responses

HTTPOS [Luo+, NDSS 2011]

Approach

Crawl Specification

Web Application

Web Crawler

FSM

Traces

Distance Metrics

Leak Quantifier

Classifier Builder

Entropy Calculator

Fisher Criterion Calculator

HTML Report

FSM of Web Application

Application States (HTML)

State Transitions (Collection of Network Traces)

Attacker Builds a Classifier to Identify State Transitions
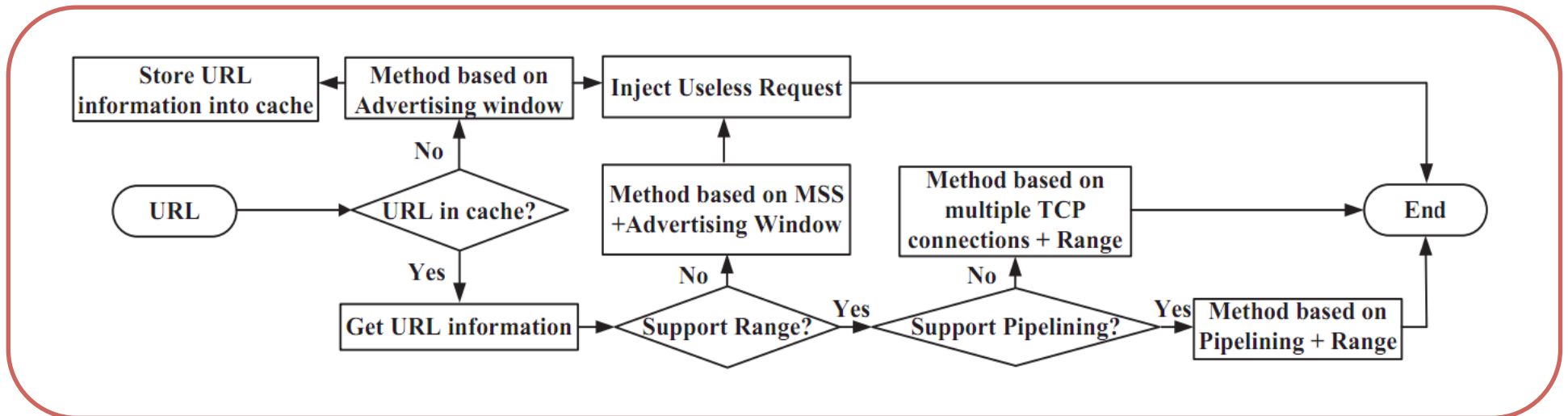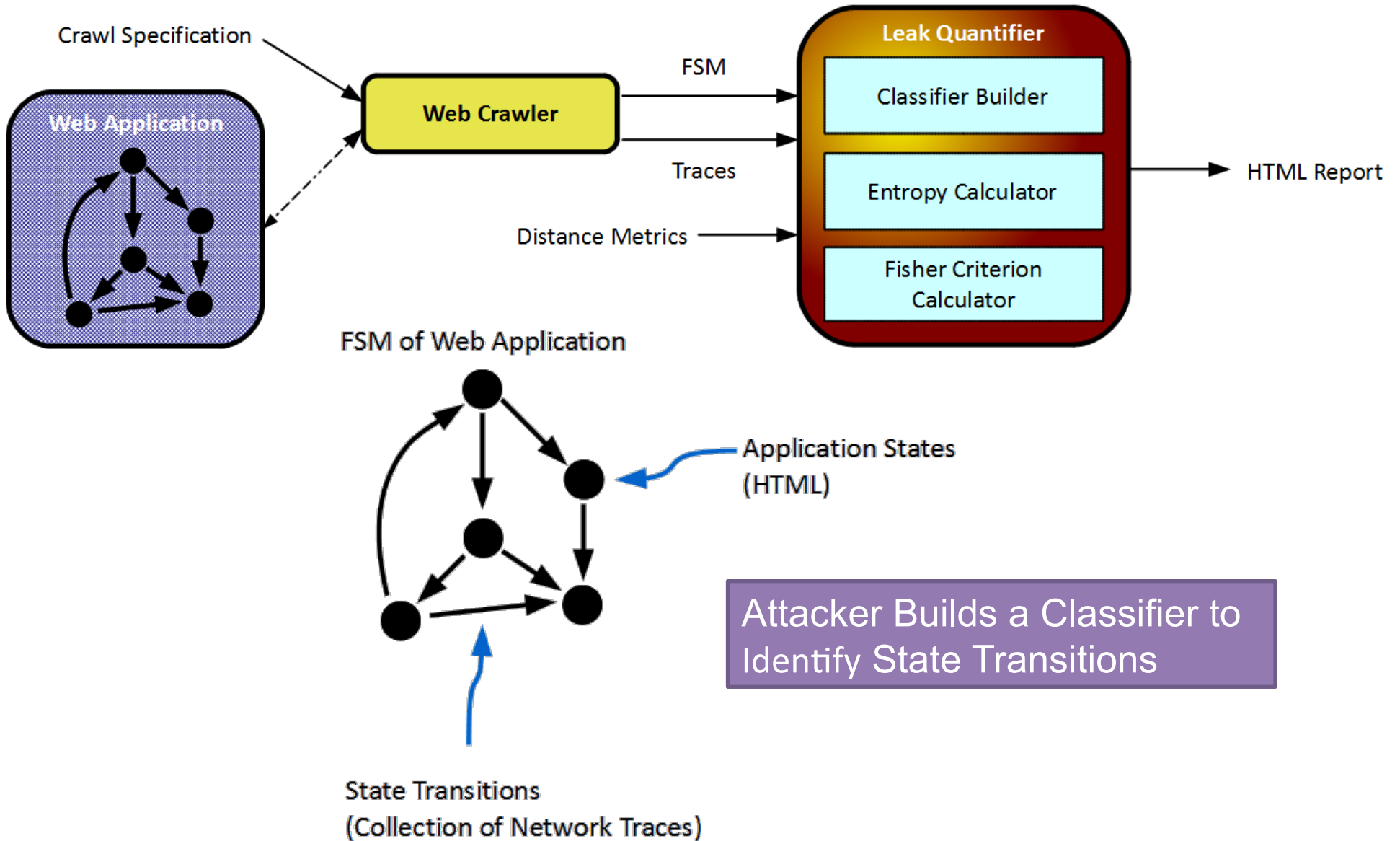
A Black-Box Approach

Similar to Real Attack Scenario

HTTPS over WPA2
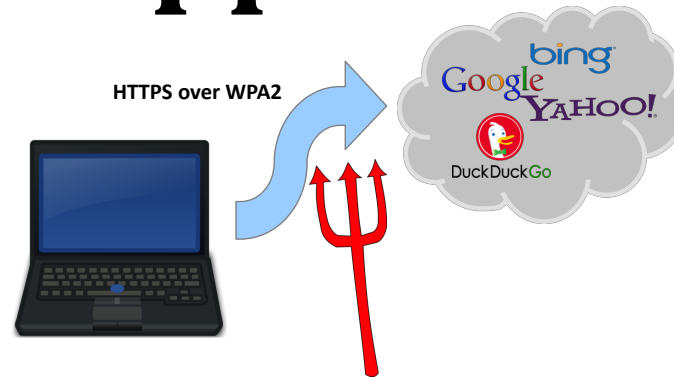
Applicable to Most Web Applications

Full Browser Analysis

Black-Box Web Application Crawling
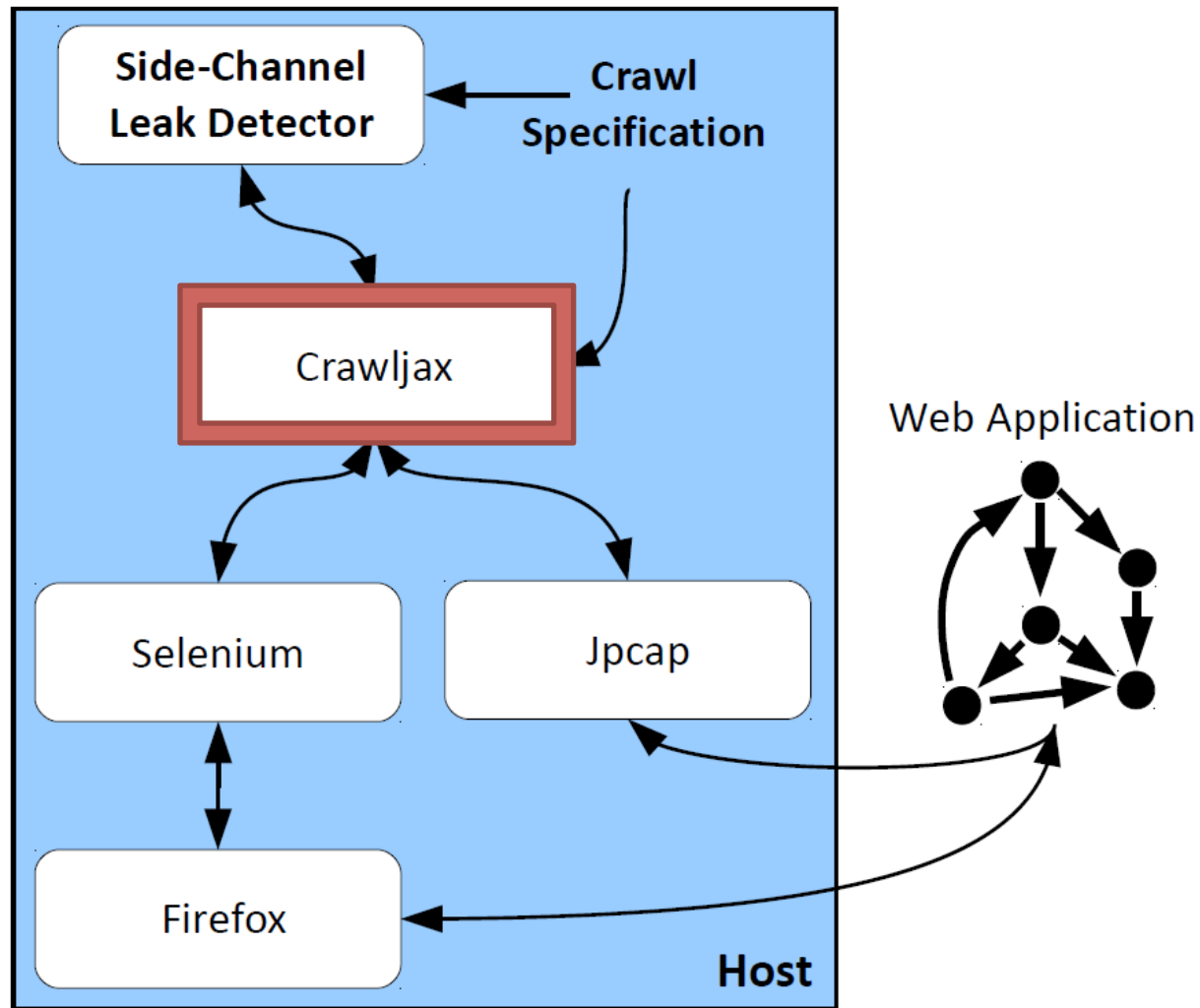
Crawljax

Crawljax
Automate Ajax Crawling and Testing

About Download Documentation Plu

Home

The Crawljax team is pleased to announce the crawljax-2.0 release. This release supports multi-browser crawling and includes many improvements.

Crawljax is an open source Java tool for automatically crawling and testing modern (Ajax) web applications.

Crawljax can crawl any Ajax-based web application by firing events and filling in form data. It creates a *state-flow graph* of the dynamic DOM states and the transitions be inferred state-flow graph forms a very powerful base for many types of auto testing:

- Invariant-based testing
- Regression testing
- Non functional testing (Accessibility, validation, I18n, security, ...)
- Detecting broken links/images/tooltips
- And via its plugin architecture many more...

Documentation

You can follow us on Twitter!

- Log in

Web crawling back-end drives Firefox instance via Selenium

Designed to build Finite-State Machines of AJAX Applications

http://crawljax.com/

Approach
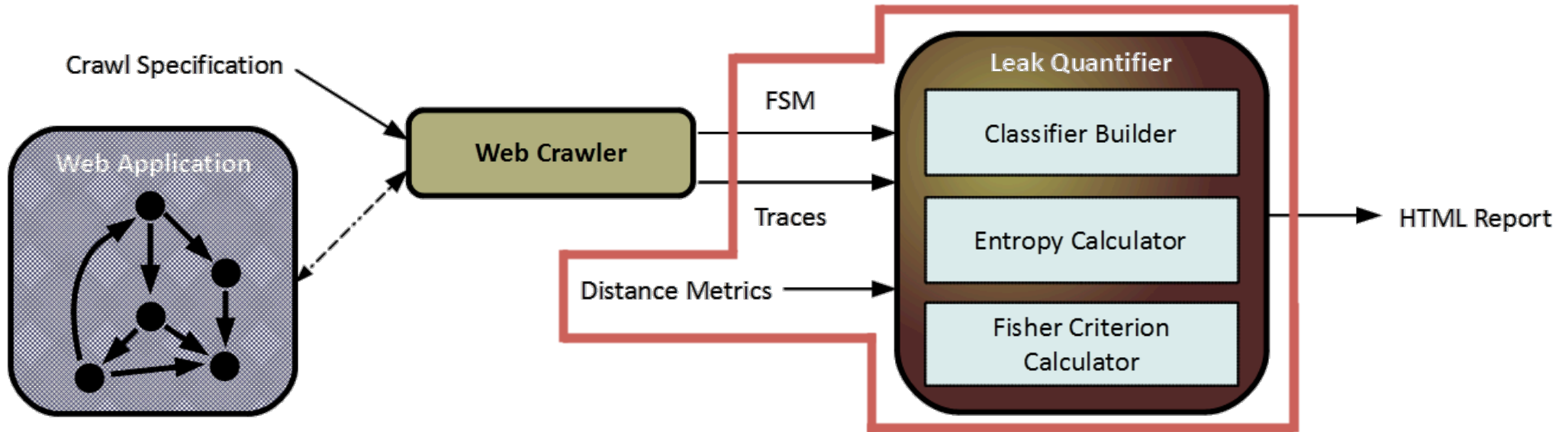
Threat Models and Assumptions

Both: Victim begins at root of application

WiFi
No disruptive traffic
Distinguish incoming and outgoing

ISP
Access to TCP header
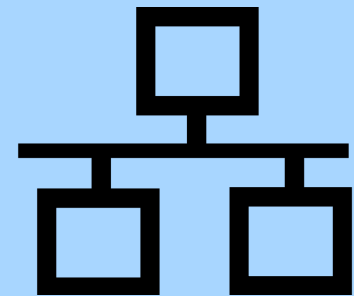
Nearest-Centroid Classifier

FSM of Web Application

Application States
(HTML)

State Transitions
(Collection of Network Traces)
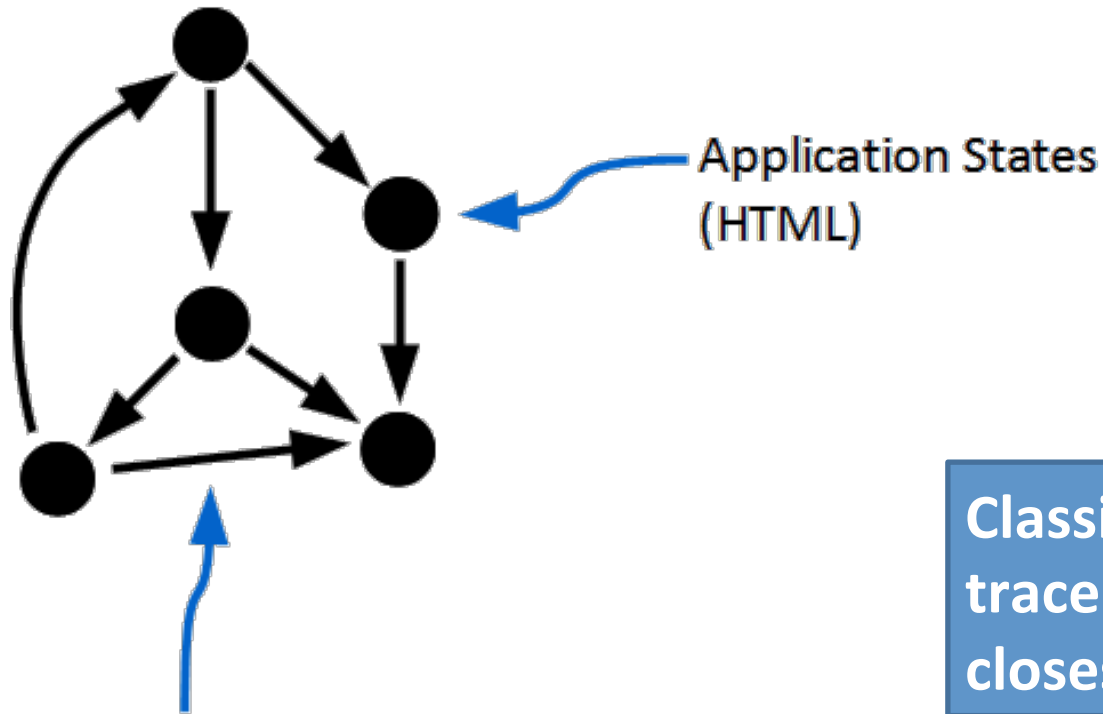
Given an unknown network trace, we want to determine to which state transition it belongs

Classify unknown trace as one with the closest centroid

Distance Metrics

Metrics to determine similarity between two traces

Size-Weighted-Edit-Distance

Convert to string, weighted edit distance based on size

Edit-Distance

Unweighted edit distance

```
                                        72.14.204 -> 192.168.1 62 bytes
                          192.168.1 -> 72.14.204 281 bytes
         192.168.1 -> 72.14.204 62 bytes        A 62 bytes
              72.14.204 -> 192.168.1 1860 bytes
         72.14.204 -> 192.168.1 62 bytes  192.168.1 -> 72.14.204 294 bytes
              192.168.1 -> 72.14.204 482 bytes        A 482 bytes
         192.168.1 -> 72.14.204 482 bytes  72.14.204 -> 192.168.1 296 bytes
         72.14.204 -> 192.168.1 693 bytes        B 693 bytes
              192.168.1 -> 72.14.204 453 bytes
         192.168.1 -> 72.14.204 62 bytes         A 62 bytes
              72.14.204 -> 192.168.1 2828 bytes
         72.14.204 -> 192.168.1 62 bytes         B 62 bytes
         192.168.1 -> 72.14.204 281 bytes        A 281 bytes
         72.14.204 -> 192.168.1 1860 bytes       B 1860 bytes
         192.168.1 -> 72.14.204 294 bytes        A 294 bytes
         72.14.204 -> 192.168.1 296 bytes        B 296 bytes
         192.168.1 -> 72.14.204 453 bytes        A 453 bytes
         72.14.204 -> 192.168.1 2828 bytes       B 2828 bytes
```

Classifier Performance – Google Search

First character typed, ISP threat model
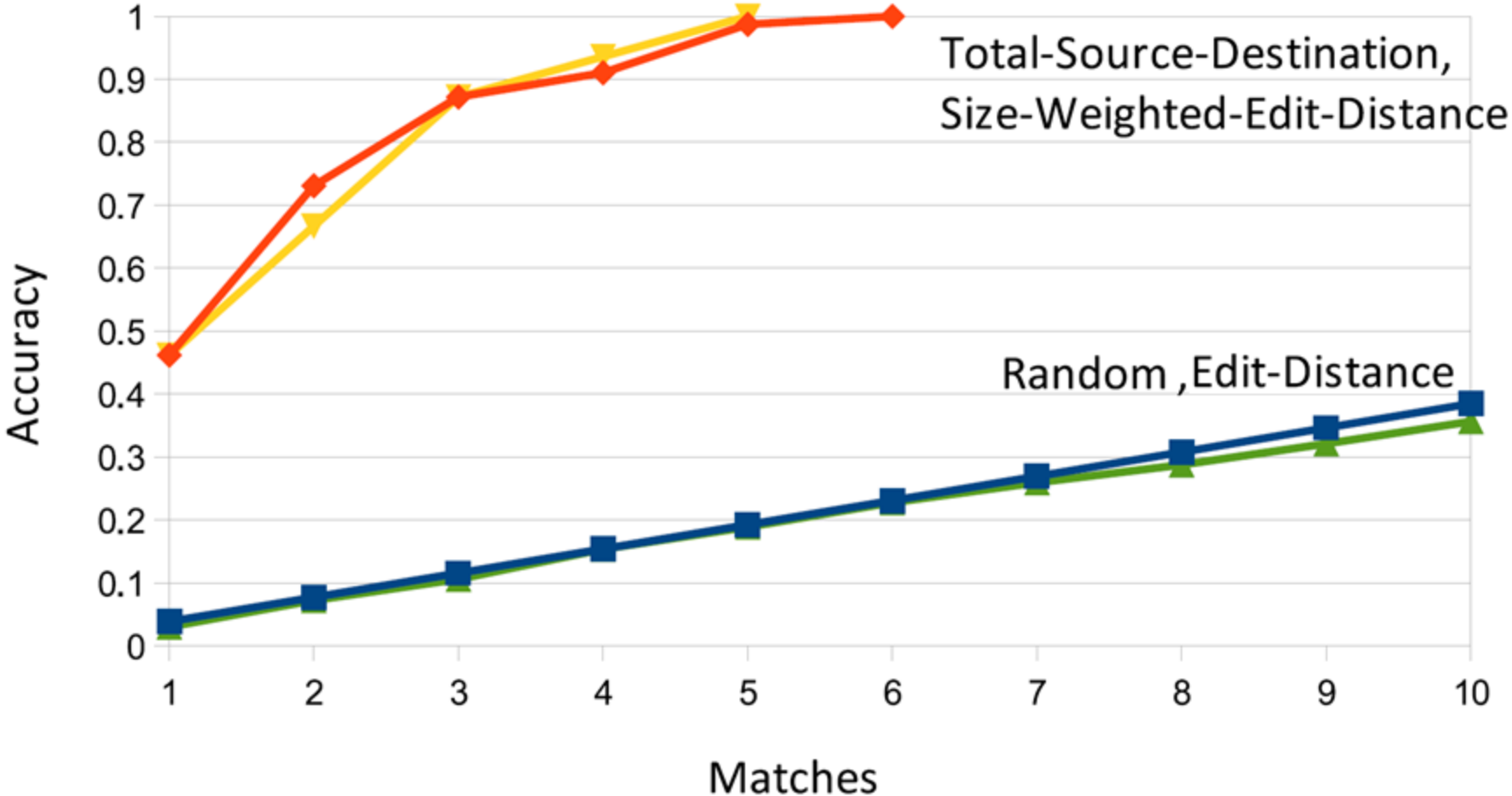
Quantifying Leaks
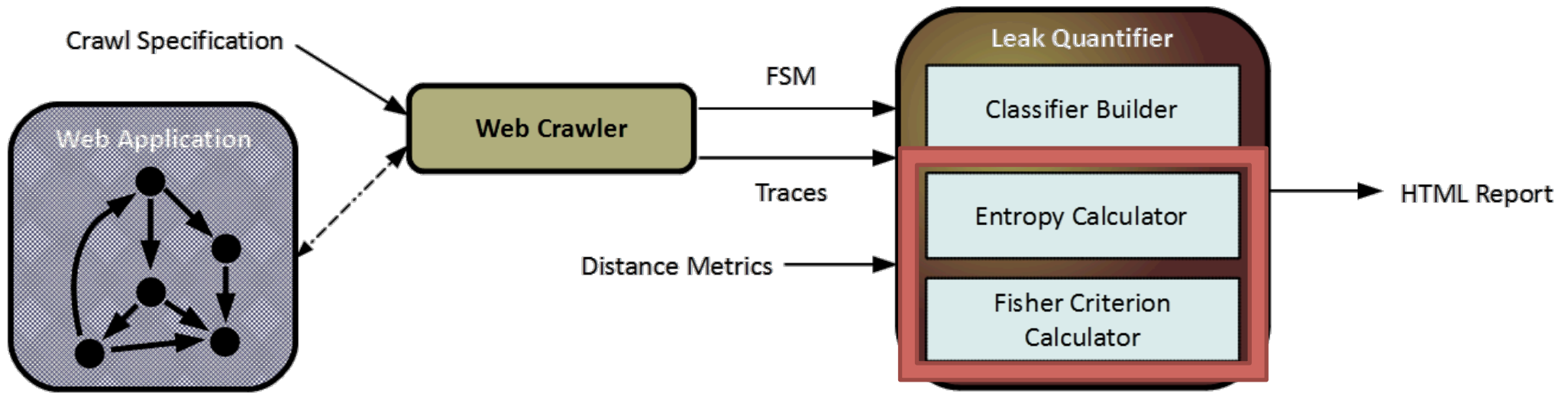
Leak quantification should be independent of a specific classifier implementation

Entropy Measurements

Entropy measurements are a function of the average size of an attacker's uncertainty set given a network trace

Problems
The same network trace can be the result of multiple classifications

Every possible network trace is unknown

Size of uncertainty set

Centroid for class

Use the centroids

$$H(X) = \sum_{i=0}^{n} \frac{\log_2 p(\bar{x}_i)}{n}$$

Number of classes

Traditional Entropy Measurement

Determining Indistinguishability

At what point are two classes indistinguishable (same uncertainty sets)?
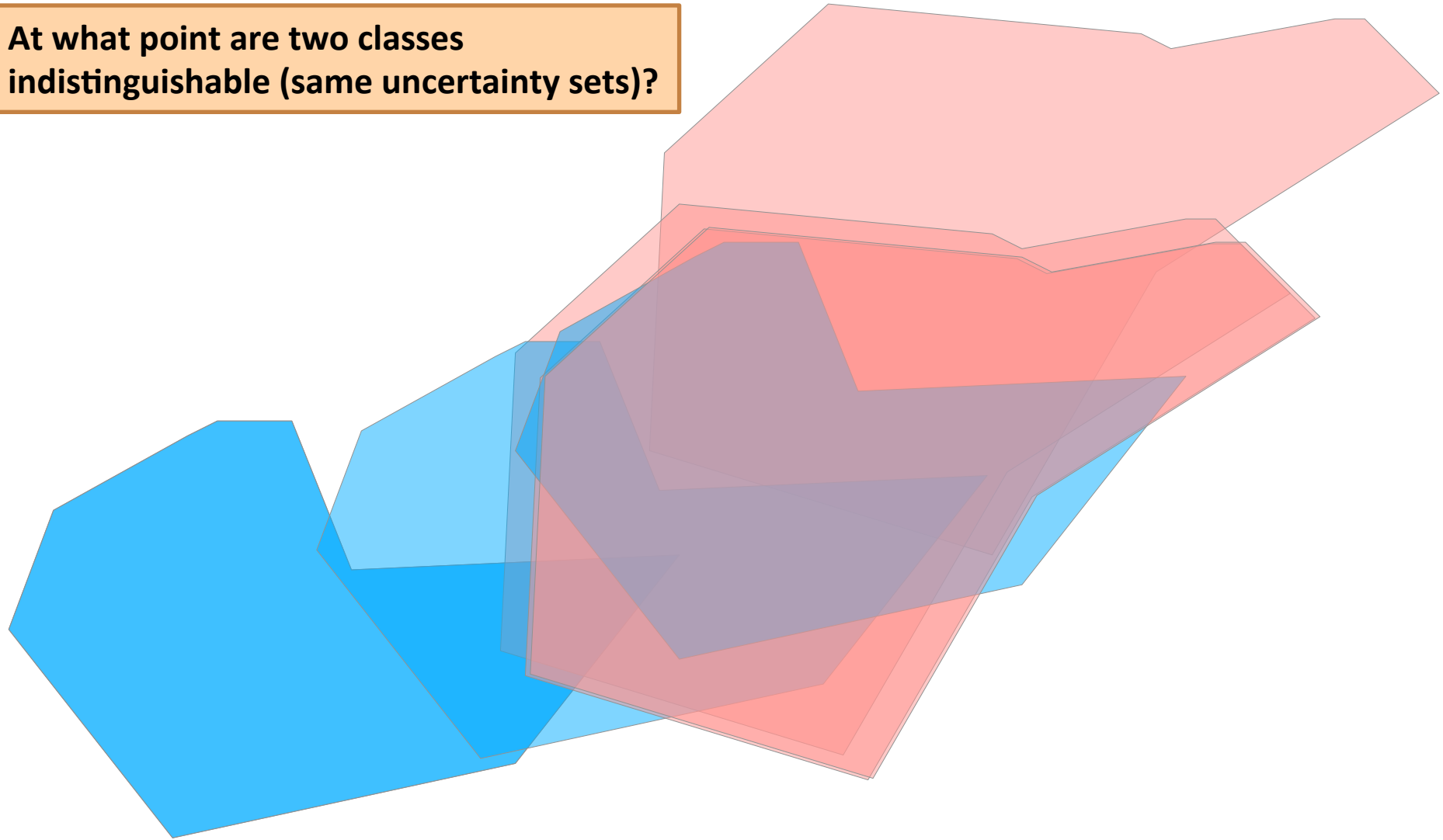
Determining Indistinguishability

Compare points to centroids?
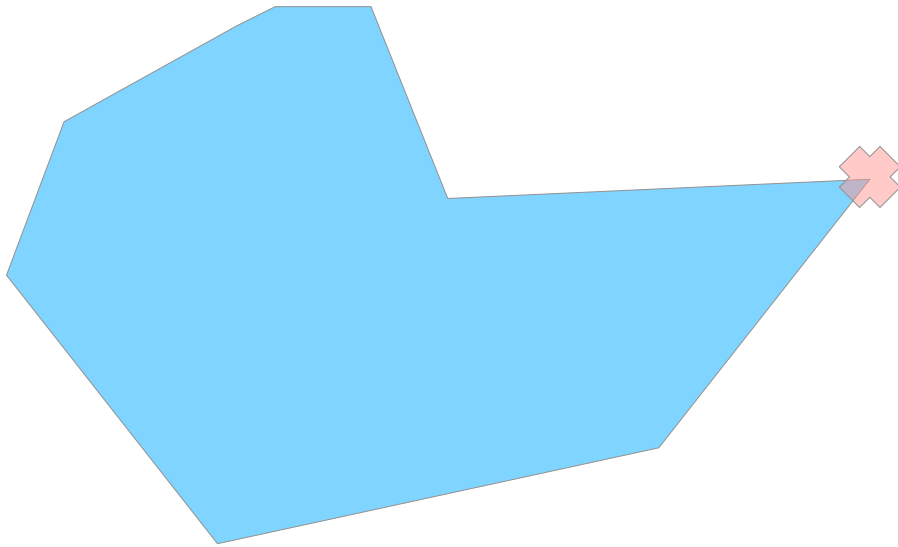
Same issue with individual points.

In practice the area can be very large due to high variance in network conditions

Entropy Distinguishability Threshold
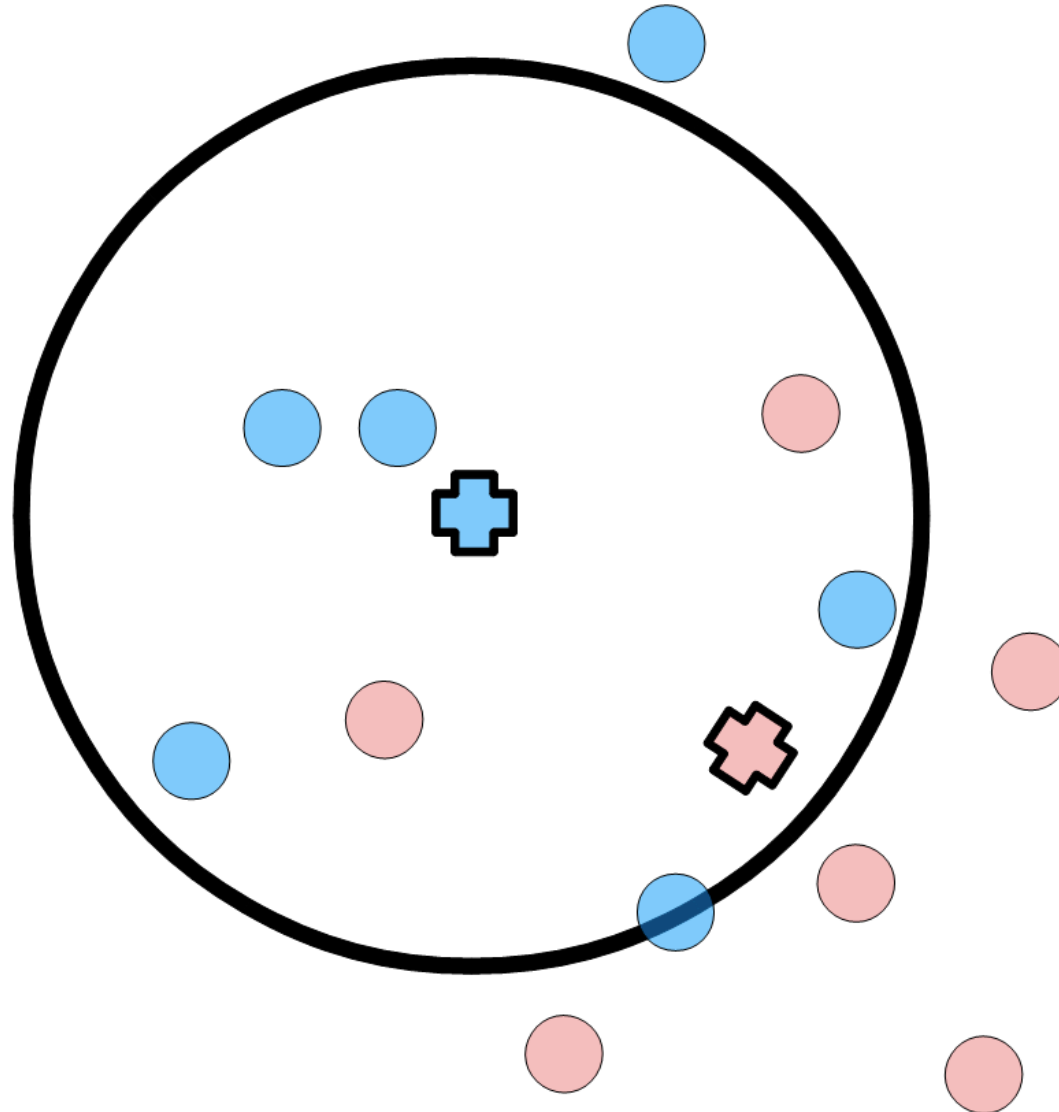
Threshold of 75%

Google Search Entropy Calculations

	Threshold		
	100%	**75%**	**50%**
Desired	4.70	4.70	4.70
Total-Source-Destination	2.95	2.40	0.44
Size-Weighted-Edit-Distance	1.13	0.56	0.44
Edit-Distance	4.70	4.70	4.70

(measured in bits of entropy)

We'd rather not use something with an arbitrary parameter

Fisher Criterion

[11] Ronald A. Fisher. The Use of Multiple Measurements in Taxonomic Problems. *Annals of Eugenics*, 1936.

Fisher Criterion
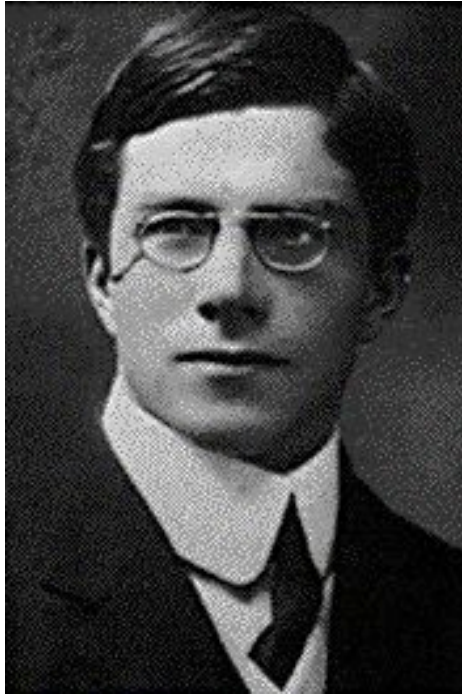
Ronald Fisher (1890-1962)

Marred Arthur Guinness' daughter, secret wedding (she was 17) in 1917

Arthur Guinness (1835-1910)

Developed many statistical tools as a part of his prominent role in the eugenics community

Fisher Criterion

Like all good stories, this one starts with a Guinness.

Arthur Guinness (1725-1803)

"Guinness is Good for You"

Fisher Criterion

$$F(X) = \frac{\sigma^2_{between}}{\sigma^2_{within}} = \frac{\sum\limits_{i=0}^{n} m_i (\bar{x}_i - \bar{x})^2}{\sum\limits_{i=0}^{n}\sum\limits_{j}^{m} (x_{ij} - \bar{x}_i)^2}$$

$F(X)=11$
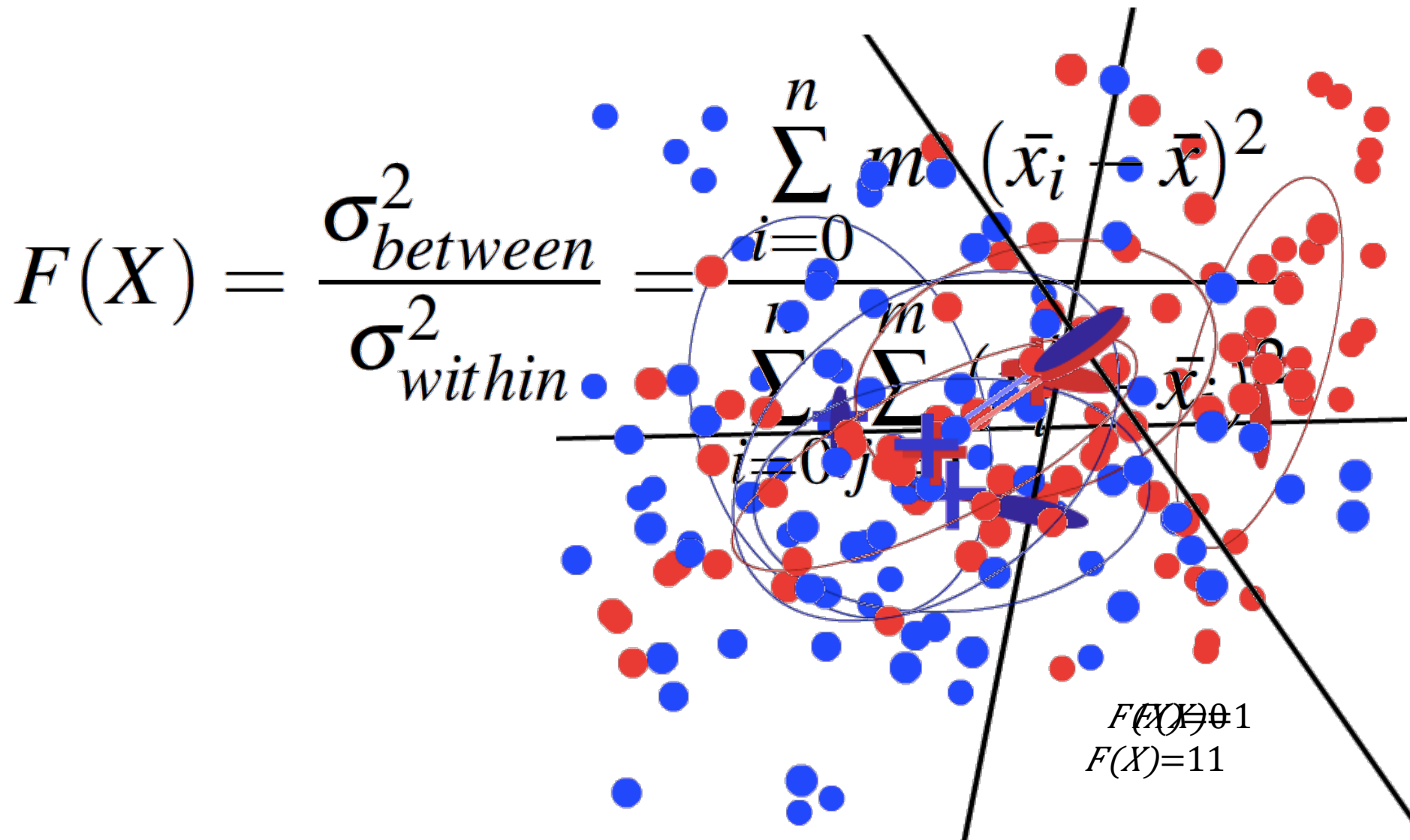
Google Search Fisher Calculations

Fisher Criterion Calculations	
Total-Source-Destination	4.13
Size-Weighted-Edit-Distance	41.7
Edit-Distance	0.00

Entropy Calculations

	100%	75%	50%
Desired	4.70	4.70	4.70
Total-Source-Destination	2.95	2.40	0.44
Size-Weighted-Edit-Distance	1.13	0.56	0.44
Edit-Distance	4.70	4.70	4.70

Other Applications

Bing Search Suggestions

Total-Source-Destination, Size-Weighted-Edit-Distance

Edit-Distance,Random

Accuracy / Matches

Yahoo Search Suggestions

Total-Source-Destination, Size-Weighted-Edit-Distance

Edit-Distance,Random
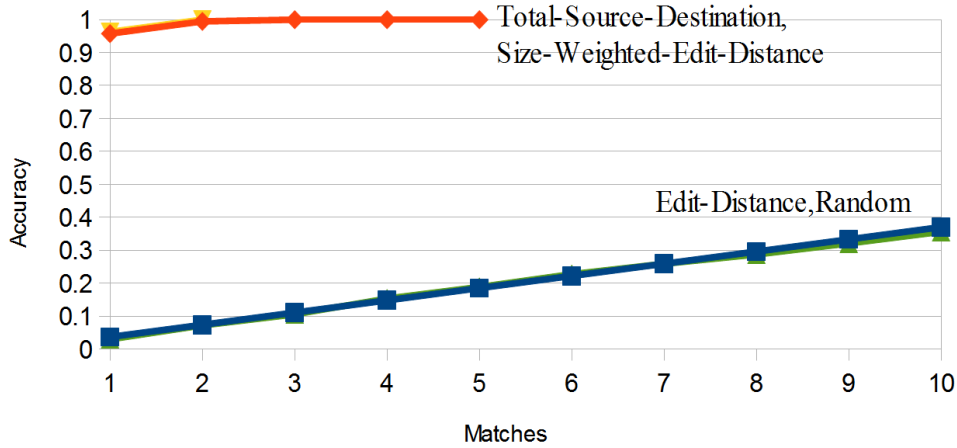
Accuracy / Matches
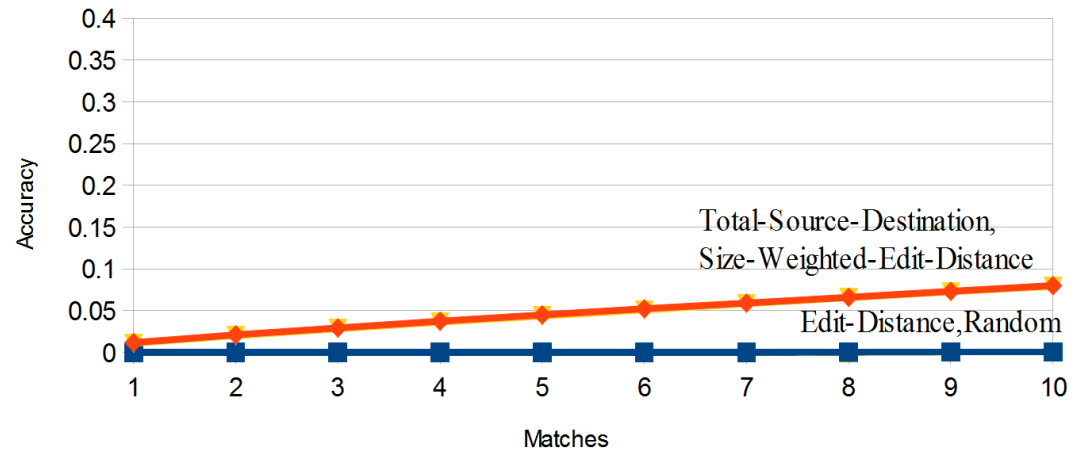
Other Applications

NHS Symptom Checker
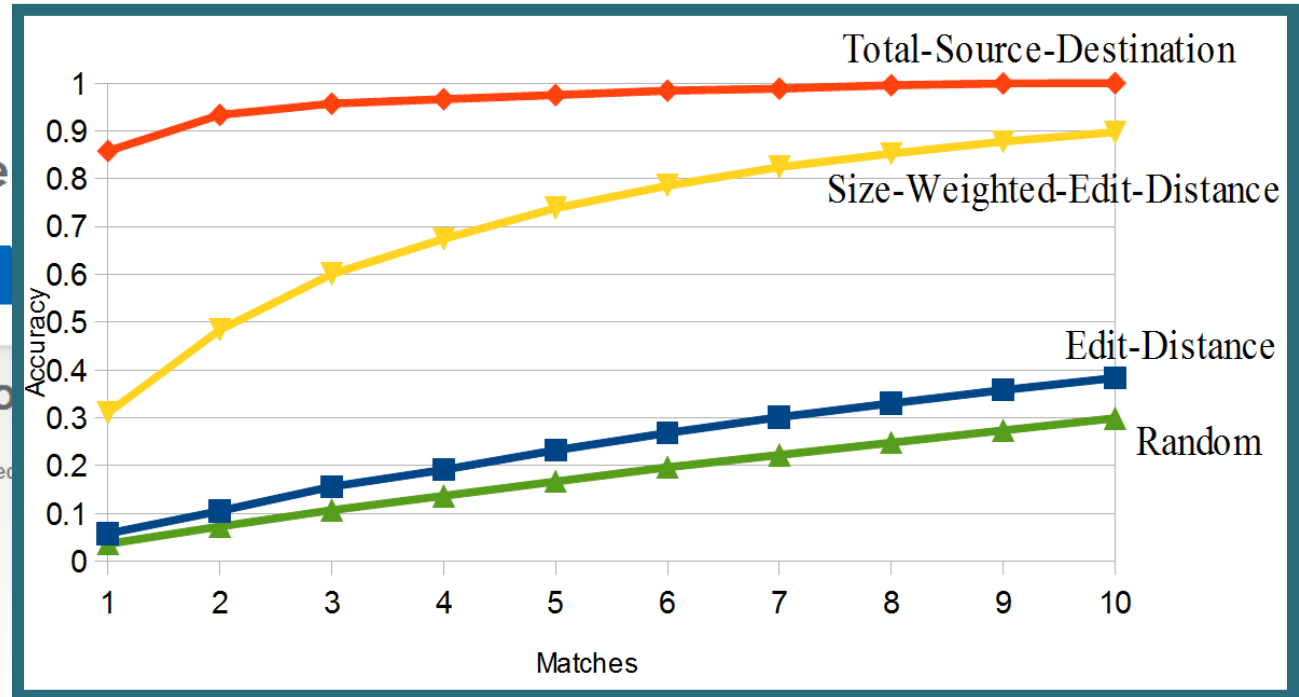
See paper for Google Health Find-A-Doctor

Evaluating Defenses

With black-box approach,
evaluating defenses is easy!

HTTPOS: Sealing Information Leaks with Browser-side Obfuscation of Encrypted Flows

Xiapu Luo[§*], Peng Zhou[§], Edmond W. W. Chan[§], Wenke Lee[†], Rocky K. C. Chang[§], Roberto Perdisci[‡]
The Hong Kong Polytechnic University[§], Georgia Institute of Technology[†], University of Georgia[‡]
{csxluo,cspzhouroc,cswwchan,csrchang}@comp.polyu.edu.hk, wenke@cc.gatech.edu, perdisci@cs.uga.edu

Abstract

Leakage of private information from web applications—even when the traffic is encrypted—is a major security threat to many applications that use HTTP for data deliv-

be profiled from traffic features [29]. A common approach to preventing leaks is to obfuscate the encrypted traffic by changing the statistical features of the packet size and packet timing inform

Existing methods for defending against information

NDSS 2011

HTTPOS Search Suggestions

(matches)

	1	10
Random	2.9%	35.6%
Total-Source-Destination	46.1%	100%
Size-Weighted-Edit-Distance	46.1%	100%
Edit-Distance	3.8%	39.5%

(matches)

	1	10
Random	2.9%	35.6%
Total-Source-Destination	3.4%	38.0%
Size-Weighted-Edit-Distance	3.8%	38.0%
Edit-Distance	3.4%	35.5%

HTTPOS Search Suggestions

Before HTTPOS

After HTTPOS

Fisher Criterion Calculations	
Total-Source-Destination	4.13
Size-Weighted-Edit-Distance	41.7
Edit-Distance	0.00

Fisher Criterion Calculations	
Total-Source-Destination	0.28
Size-Weighted-Edit-Distance	0.43
Edit-Distance	0.14

HTTPOS works well with search suggestions

HTTPOS Google Instant

(matches)

	1	10
Random	2.9%	35.6%
Total-Source-Destination	47.5%	88.3%
Size-Weighted-Edit-Distance	7.3%	52.6%
Edit-Distance	7.7%	56.0%

(matches)

After HTTPOS

	1	10
Random	2.9%	35.6%
Total-Source-Destination	43.7%	87.6%
Size-Weighted-Edit-Distance	8.2%	51.4%
Edit-Distance	8.7%	55.0%

HTTPOS Google Instant

Before HTTPOS

After HTTPOS

Fisher Criterion Calculations	
Total-Source-Destination	1.13
Size-Weighted-Edit-Distance	0.34
Edit-Distance	0.22

Fisher Criterion Calculations	
Total-Source-Destination	0.60
Size-Weighted-Edit-Distance	0.55
Edit-Distance	0.47

No training phase, so HTTPOS works well with search suggestions, but not entire pages

Summary

Developed Fisher Criterion as an alternative measurement for information leaks in this domain

Evaluated real web apps and a proposed defense system

With a tutorial

Code available now: http://www.cs.virginia.edu/sca