Protecting Web Contents against Persistent Crawlers

M.S. Thesis Defense

Department of Computer Science

The College of William and Mary

Presented by: Shengye Wan

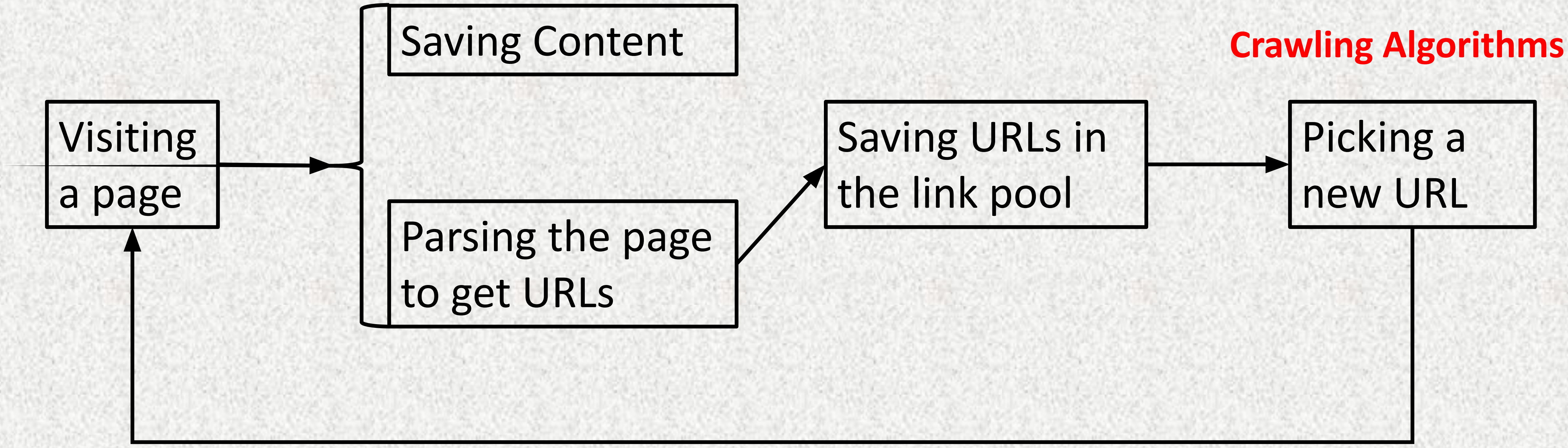
Outline

- Background
- Threat Model
- Related Work
- Our Solution
- System Design
- Experiment
- Discussion & Limitation
- Conclusion

Web Crawler

- Internet bot, systematically browses a website
- Usage: web scraping
- Copying all the pages they visit for later processing
- Consuming resources on the systems they visit

Web Crawler Workflow



Outline

- ~~Background~~
- Threat Model
- Related Work
- Our Solution
- System Design
- Experiment
- Discussion & Limitation
- Conclusion

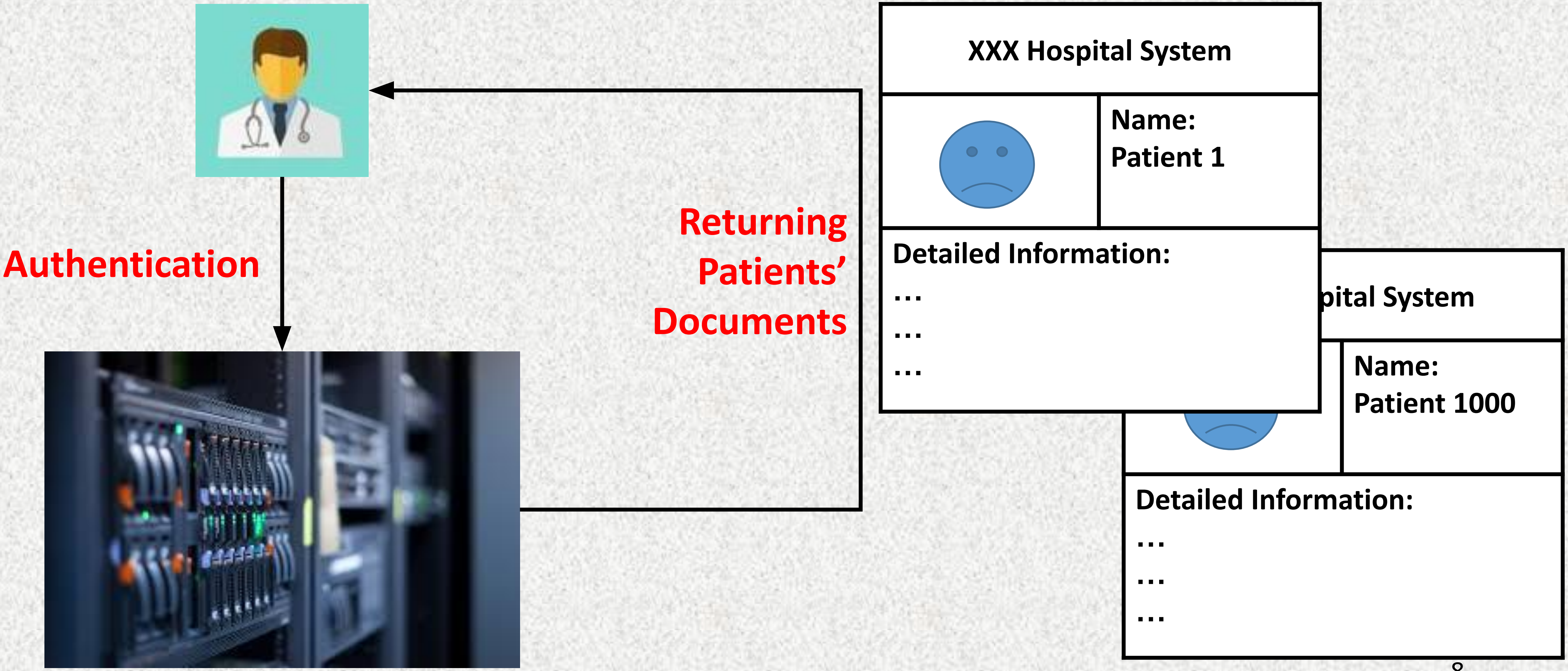
Attack of Crawler

- Stealing content
  - Phishing
    - Phishing is the attempt to acquire sensitive information of users by masquerading as a trustworthy entity in an electronic communication.
  - Putting in market

Threat Model

- Targeted website
  - Requiring users to login for viewing protected data
- Insider attacker (has legitimate user account)
- Attacker is persistent and stealthy
- Distributed crawlers
  - The total number of workers is limited

An Example of Attack



Outline

- ~~Background~~
- ~~Threat Model~~
- Related Work
- Our Solution
- System Design
- Experiment
- Discussion & Limitation
- Conclusion

Anti-Crawler Mechanism

- Several goals
  - Detecting attackers (IP, Accounts ID)
  - Suppressing & misleading attackers
  - Protecting documents from stealing by attackers

Crawler Detection Techniques

- Different solutions for defending crawlers
  - Request-related features
  - Timing-based features
  - Page-based features (popularity)
  - Clickstream related features

- Two types of detection
  - Heuristic detection & machine learning based detection

Heuristic Detection Overview

- User-Agent, referrer, visiting rate and cookie fields in the HTTP request headers
- Effective on filtering basic crawlers
- Reducing crawlers’ download efficiency

- Cannot detect all stealthy crawlers
  - Most heuristic detection features can be spoofed by attackers

Machine Learning Based Detection Overview

- In one of the earliest work, ***Discovery of web robot sessions based on their navigational,*** authors develop 24 features to train the anti-crawling model

- Most recently works are combining both detections together

- Using different sets of features

Challenges

- When
  - Crawlers are persistent

- Crawlers could sacrifice the efficiency
- Crawlers could run extra work to better mimic the access behaviors of real users
- Several insiders may coordinate

- Previous features are not good enough to stop them in the early stages

Outline

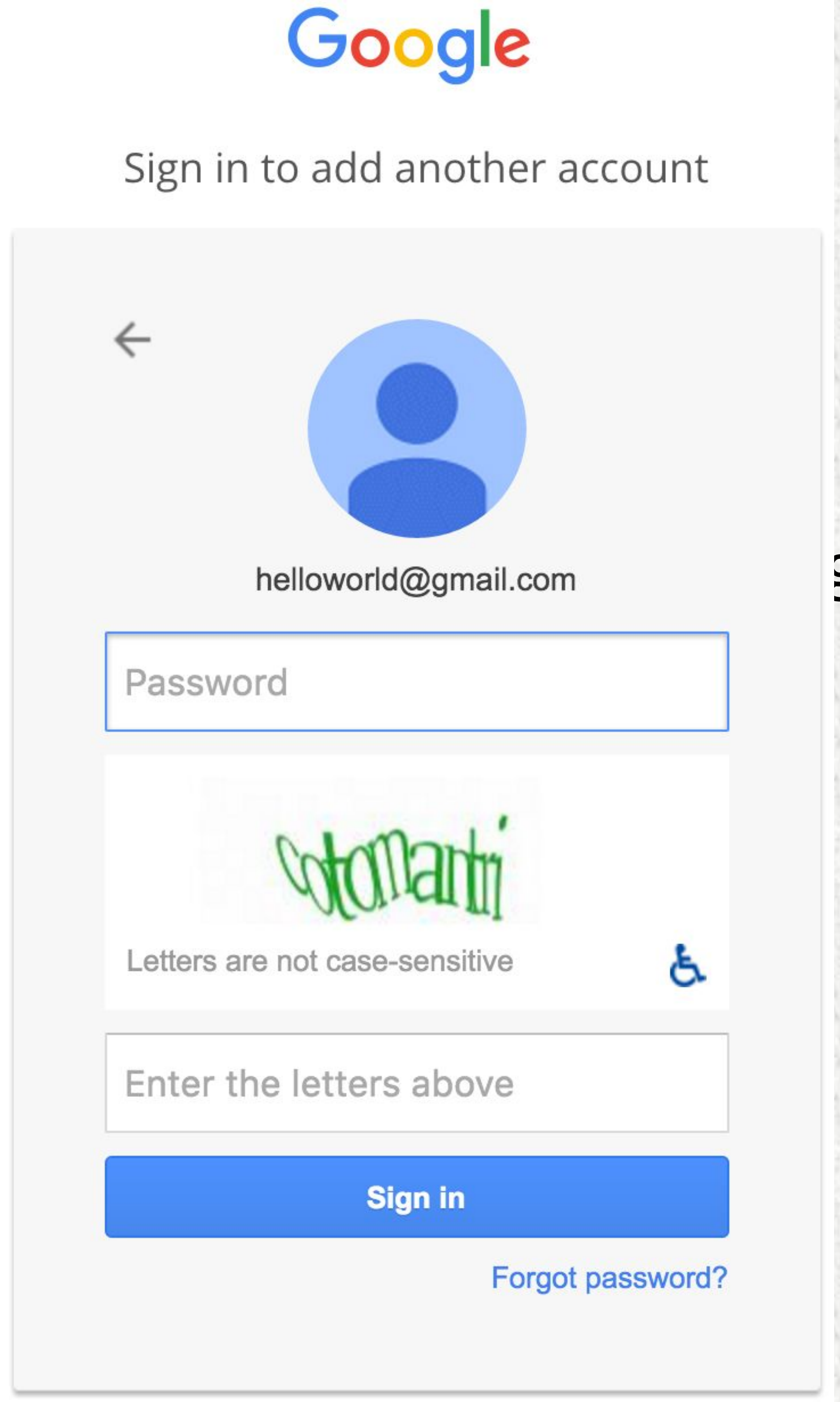
- ~~Background~~
- ~~Threat Model~~
- ~~Related Work~~
- Our Solution
- System Design
- Experiment
- Discussion & Limitation
- Conclusion

Final Goals

- High accuracy
  - Low false negative rate
  - Detecting distributed crawlers
- Fast detection **Fast is not regarding to time**
  - Stopping attacker before he or she gets too much content
- Low user experience degradation
- Delaying the crawler who could hide from mechanism
  - Suppressing the crawling efficiency to the level of human beings

Basic Architec

- Detection
  - Heuristic detect … d detection
  - Analyzing a gro … group is called one ***session***)



- Verification
  - CAPTCHA

- This architecture is … work

- Choosing features

Choosing Features

- Feature should not be spoofed by the system
- Feature should be noticed in the crawlers’ early stages

- What has not been explored very well?

- Path-based features
  - Depth & width of one user

Path-Based Features

- Used before
  - Input
    - A group of access logs (a session)
  - Output
    - Depth of this session
    - Width of this session
- Example:
  - A.com/B/C.html, depth:3, width:1
  - A.com/B/D/E.html, depth:4, width:2
- Not being used well
  - Inaccurate due to simple methodology

Path-Based Features

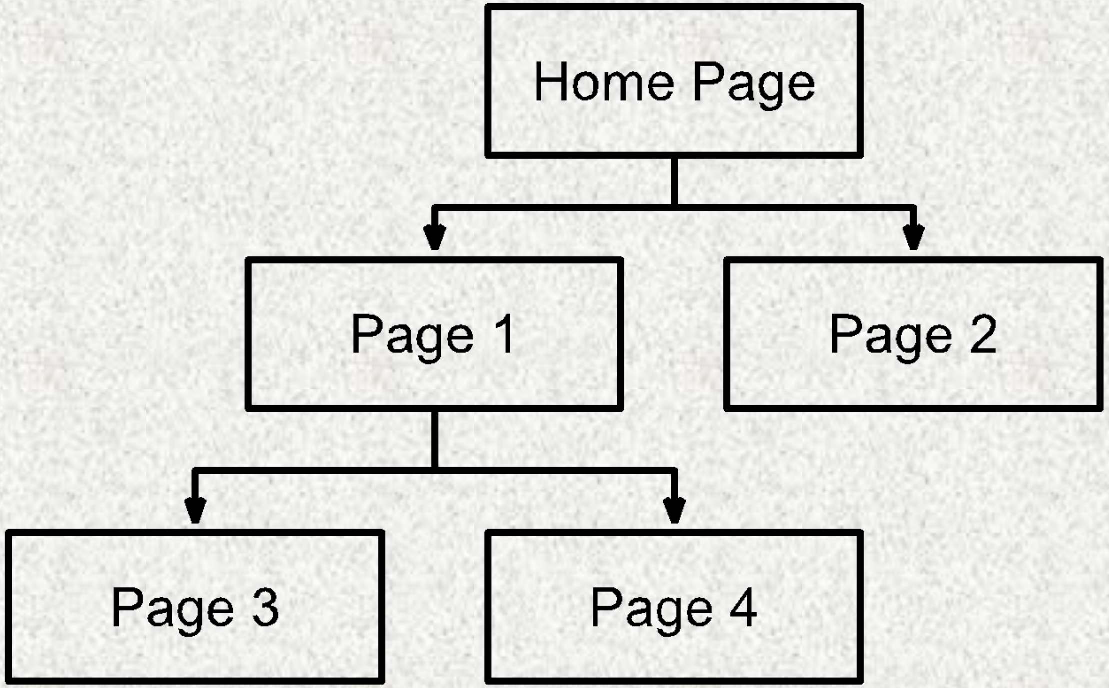
- Processing a session
  - Log by log

Definition: parent page
If USER gets page 2’s link from page 1 then page 1 is the parent page of page 2

- If one page’s parent page is viewed prior to the page within the session
  - Depth
    - Page’s depth = parent page’s depth + 1
    - Session’s depth = max(page’s depth)
  - Width
    - Parent page’s width = parent page’s width + 1
    - Session’s width = max(page’s width)

Example about Depth and Width

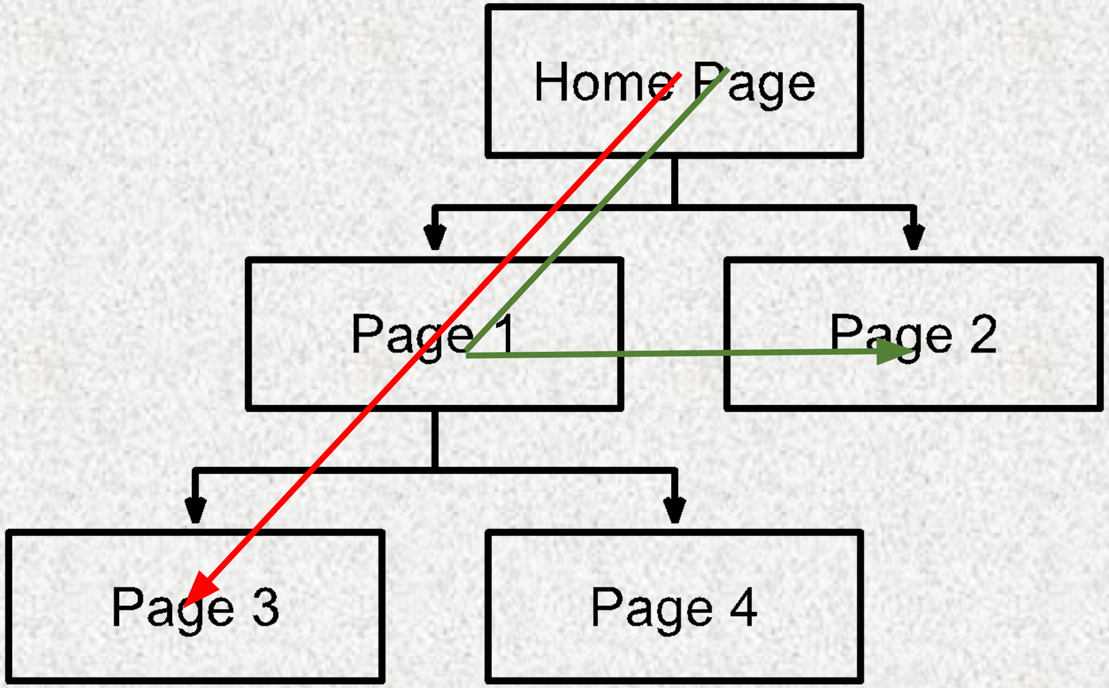
- homepage, page1, page3, page4, and page2



Path	MAX Depth	MAX Width
Home Page	1	0
Page 1	2	1
Page 3	3	1
Page 4	3	2
Page 2	3	2

Example about Depth and Width

- homepage, page1, page3
- homepage, page1, page2



Path	MAX Depth	MAX Width
Home Page	1	0
Page 1	2	1
Page 3	3	1

Depth-first

Path	MAX Depth	MAX Width
Home Page	1	0
Page 1	2	1
Page 2	2	2

Width-first

Path-Based Features Observation

- Crawlers are working based on crawling algorithms
  - We could classify them into three types
  - Depth-first, Width-first and random-like (like PageRank-first)

- Human have their patterns regarding to path’s depth or width
  - Short term
    - Either Depth-first or Width-first
  - Long term
    - No certain pattern

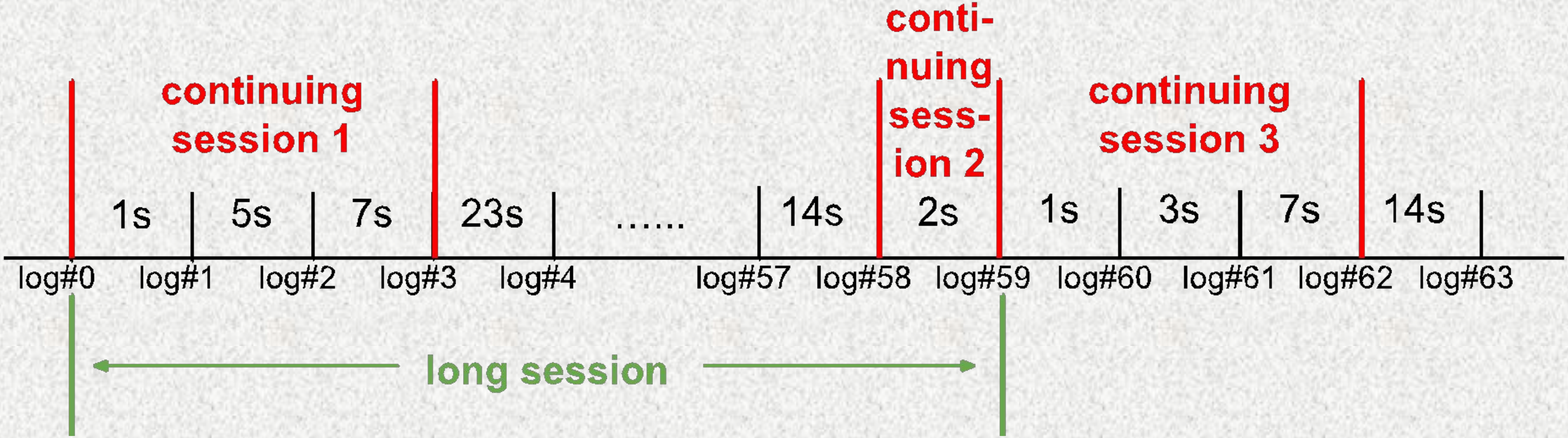
New Concepts

- Continuing Session (short term)
  - A session that describes user continuous access behavior
  - User requests two pages within a couple of seconds
  - We use 10 seconds as the default interval time
  - Time gap can be tuned according to each website's specific user scenario
  - Length of a continuing session varies depending on the visiting pattern of the users

New Concepts

- Long Session (long term)
  - A session that describes user general access behavior
  - Length of a long session is fixed
    - If length is X, then first long session = first X access logs
  - A long session's length is suggested as twice of the average length of continuing sessions
  - A continuing session only belongs to one long session

Novel Conceptions



Features For Machine Learning

max(

L_{Lo}

Depth rate of long session

max(D_L) represents the maximum visiting path depth in a long session

L_{Long} represents the fixed length of the long session

max(

L_{L}

Width rate of long session

max(D_L) represents the maximum visiting path width in a long session

Features For Machine Learning

$$\left|\frac{\max(D_L)}{L_{Long}} - \frac{\max(D_C)}{L_{Contin}}\right|$$

The absolute difference between
depth rate of long session and
depth rate of longest continuing session
in this long session

$$\left|\frac{\max(W_L)}{L_{Long}} - \frac{\max(W_C)}{L_{Contin}}\right|$$

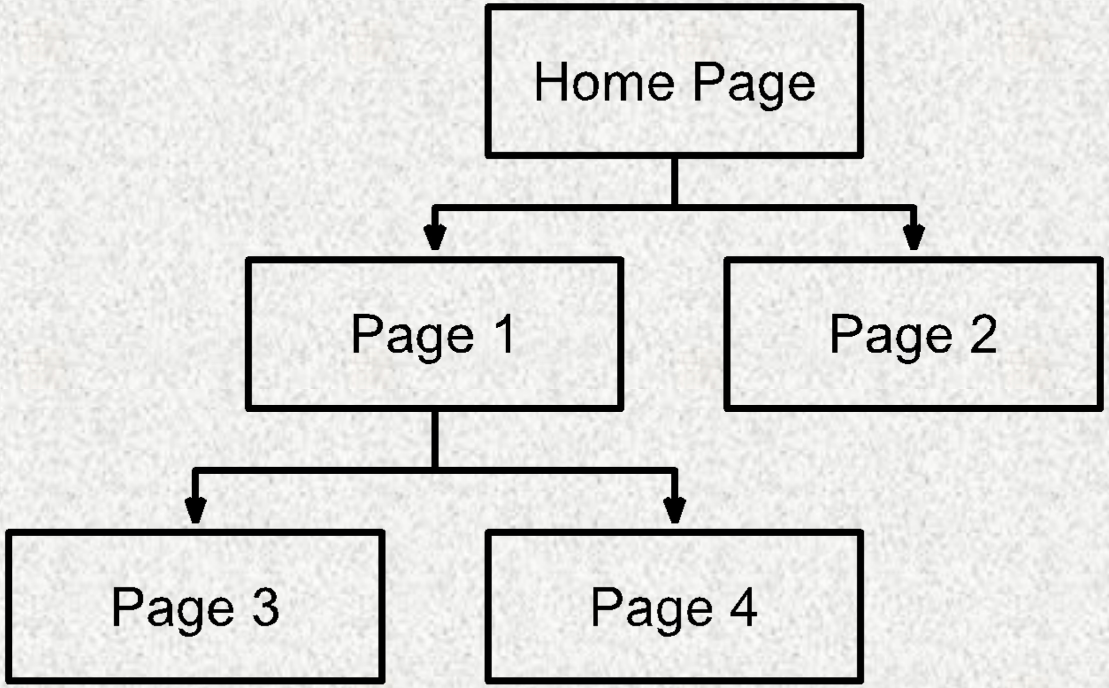
The absolute difference between
width rate of long session and
width rate of longest continuing session
in this long session

Calculating Depth and Width

- Current situation
  - We only have sessions of access logs
  - We do not know the parent page of every log's link

Difficulty in Calculating Depth and Width

- homepage, page1, page3, page4, and page2



Assumption:
we know every log's parent page

Path	MAX Depth	MAX Width
Home Page	1	0
Page 1	2	1
Page 3	3	1
Page 4	3	2
Page 2	3	2

What if both page1 and page 3 contain page4's link

Calculating Depth and Width

- Current situation
  - We only have sessions of access logs
  - We do not know the parent page of every log’s link
- What we want to get
  - Accurate depth and width
- Solution
  - Adding marker to every URL
  - Markers include parent pages’ URL and parent page’s obtainer

Adding Marker

- A typical URL of the domain A is: A.com/B/C.html

- After we add the URL marker to it, it would be: A.com/B/C.html/mk:B/root.html;User1

- Appended URL marker is mk:B/root.html;User1
  - This URL is retrieved from the page A.com/B/root.html
  - "User1" is the user who obtains the URL

- The whole URL after encryption using AES-256-CBC: A.com/en:bf37cf8f8f6cb5f3924825013e3f79c04086d1e569a7891686fd7e3fa3818a8e

Adding Marker Example

Log Info of User1			Analyzing Result					
URL	Marker	timestamp	Continuing Session ID	Deepest page	MAX depth	Widest Page	MAX width
URL1	URL0;1	0	1	URL1	1	URL0	0
URL2	URL1;1	3	1	URL2	2	URL1	1
URL3	URL2;1	8	1	URL3	3	URL1	1
URL2	URL3;1	10	1	URL2	4	URL1	1
URL4	URL2;1	15	1	URL4	5	URL2	2
URL5	URL2;1	17	1	URL4	5	URL2	3
URL6	URL3;3	20	1	URL4	5	URL2	3
URL7	URL1;1	32	8	URL4	5	URL2	3

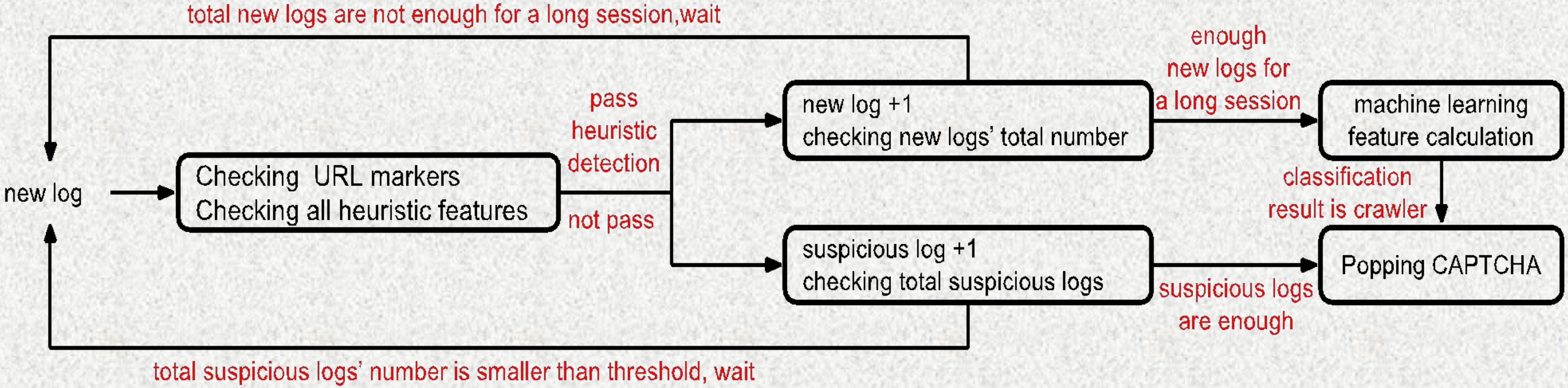
Benefit of Marker

- Reliable Information
  - Calculation is accurate
  - Marker cannot be forged by attacker
- Misleading Crawlers
  - One page has different markers and thus different URLs
    - Different Parent pages
    - Different Users
- Defending distributed crawlers

Outline

- ~~Background~~
- ~~Threat Model~~
- ~~Related Work~~
- ~~Our Solution~~
- System Design
- Experiment
- Discussion & Limitation
- Conclusion

Working Process



PathMarker Working Process

Heuristic Detection Design

- Page visiting rate, referrer, user agency, and cookies
  - If one or more of these fields in over 10 HTTP requests of one user within an hour are abnormal, label the user as a potential crawler

- URL marker integrity checking
  - Decrypting the URL marker
  - Comparing the visitor of this page with the one recorded in the URL marker (who is the obtainer of the page URL)

Machine Learning Design

- SVM
  - Support vector machines
  - Supervised learning
  - Providing both one-class SVM and Multi-class SVM

- 6 Features(4 has been discussed before)

$$\frac{\max(D_L)}{L_{Long}} \qquad \frac{\max(W_L)}{L_{Long}}$$

$$\left|\frac{\max(D_L)}{L_{Long}} - \frac{\max(D_C)}{L_{Contin}}\right| \qquad \left|\frac{\max(W_L)}{L_{Long}} - \frac{\max(W_C)}{L_{Contin}}\right|$$

Machine Learning Design

$$\frac{\mathrm{Var}(\ldots}{\overline{I_L}^2}$$

I_L is the time gap between two consecutive requests of a long session

This feature is computed as the
variance of time interval in a long session over
the square of the average time interval in the long session

$$\frac{\mathrm{Var}(\ldots}{\overline{I_C}^2}$$

I_C means the time interval in a continuing session

Outline

- ~~Background~~
- ~~Threat Model~~
- ~~Related Work~~
- ~~Our Solution~~
- ~~System Design~~
- Experiment
- Discussion & Limitation
- Conclusion

Experiment Setup

- Building a student online forum
  - Collecting data from one month period
  - Using 6 types of crawlers to crawl the forum
  - Half training and half testing
  - Case study – Google bots
- Running simulation on efficiency degradation of distributed crawler introduced by Markers

Real Data Classification Result Table

Type 0, normal users
Type 2, Depth-first

Type 1, Width-first
Type 3, random-like

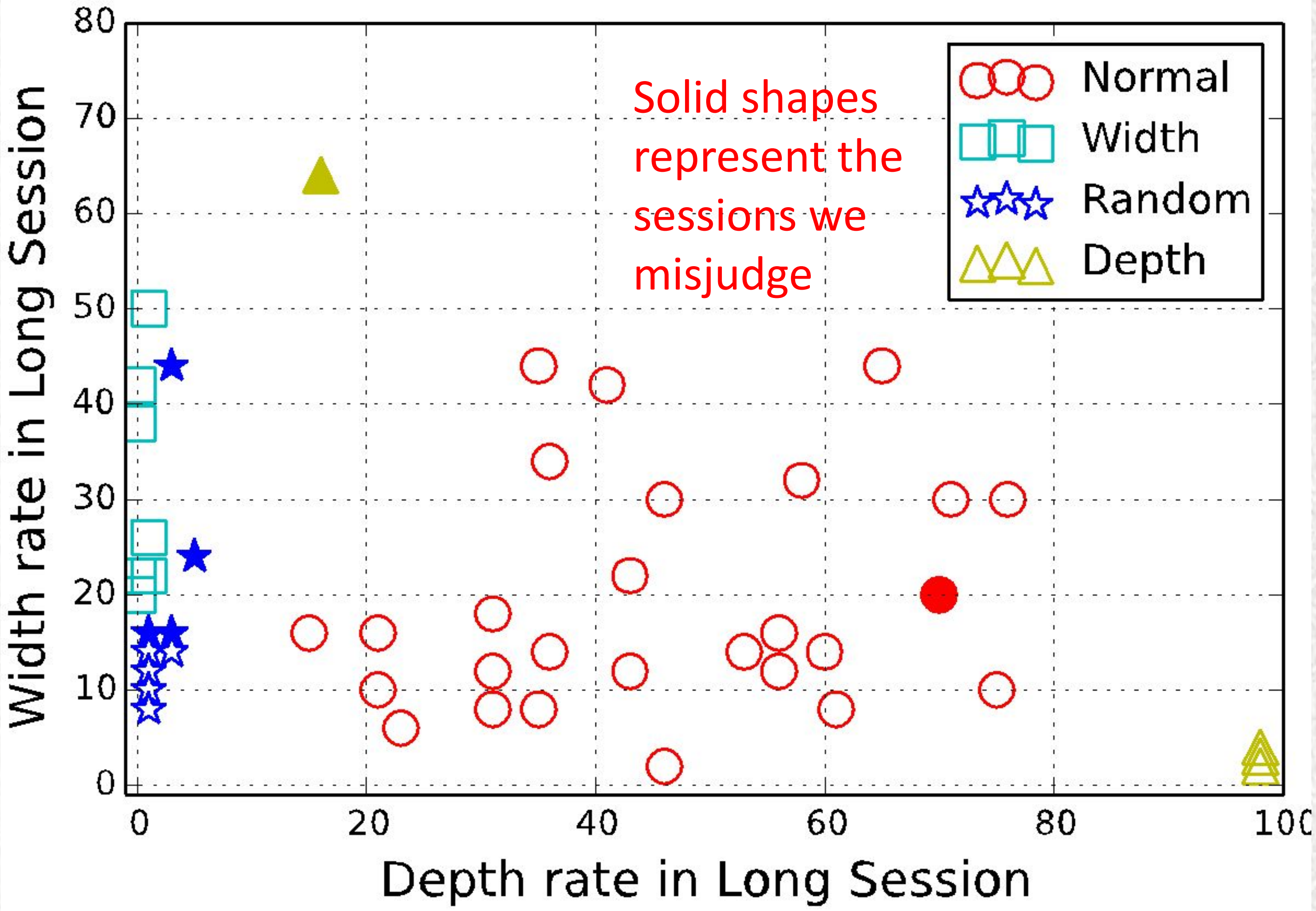
Original Type	Classify As 0	Classify As 1	Classify As 2	Classify As 3
0	96.43%	0%	3.57%	0%
1	0%	100%	0%	0%
2	0%	6.25%	93.75%	0%
3	1.51%	1.77%	0%	96.72%

The only false negative case: we misjudge crawlers as normal users

* There is at least one other long session of the same crawler that implies the visitor is not a human being so in fact we do not miss any crawler

WILLIAM & MARY

Real Data Classification Result 1

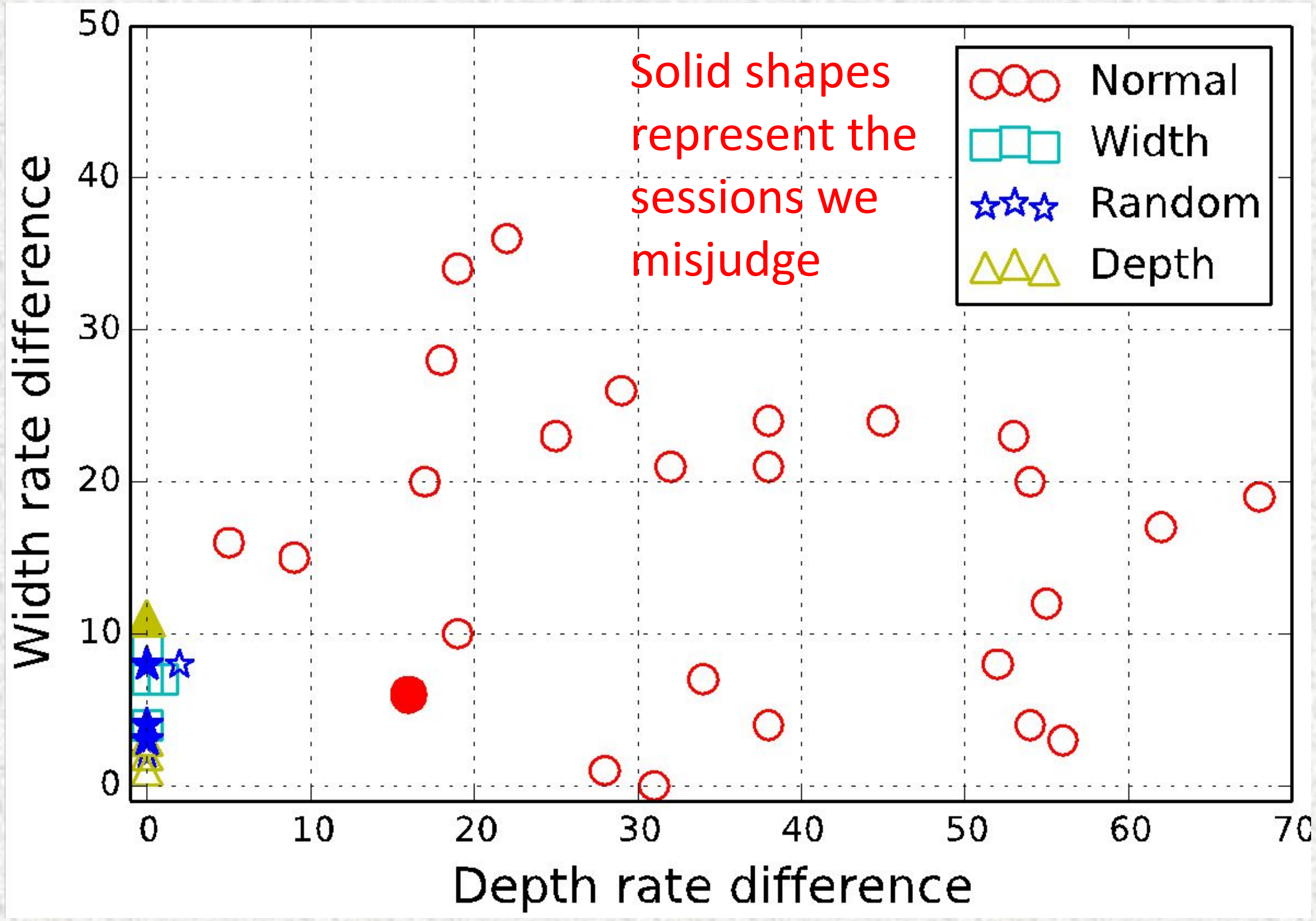


Differences Between Crawlers and Users about feature 1 and 2

$$\frac{\max(}{L_{Lo}}$$

$$\frac{\max(}{L_{L}}$$

Real Data Classification Result 2

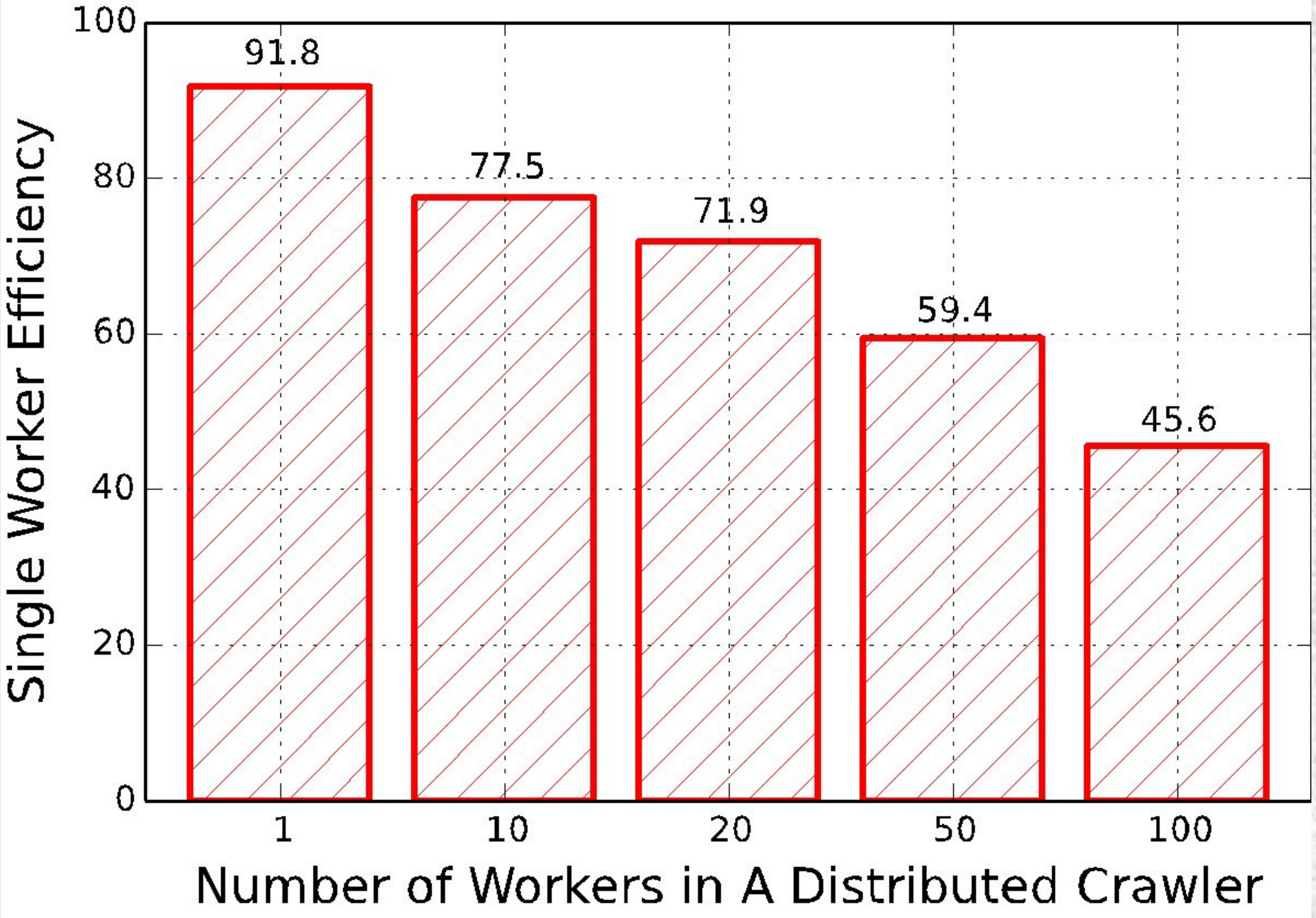


Differences Between Crawlers and Users about feature 4 and 5

$$\left|\frac{\max(D_L)}{L_{Long}} - \frac{\max(D_C)}{L_{Contin}}\right|$$

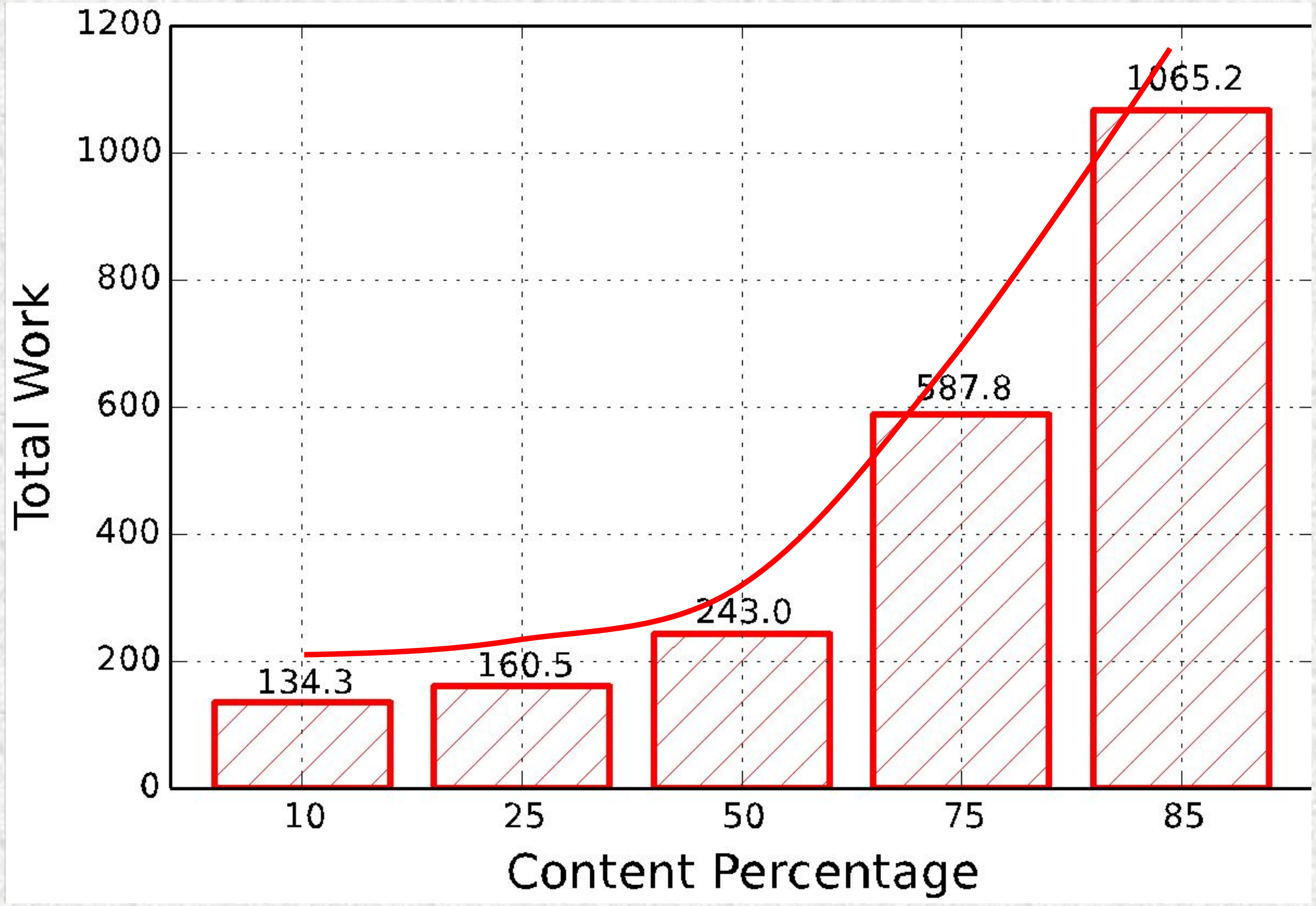
$$\left|\frac{\max(W_L)}{L_{Long}} - \frac{\max(W_C)}{L_{Contin}}\right|$$

Simulation for Distributed Crawlers 1



Suppressing Distributed Crawlers

Simulation for Distributed Crawlers 2



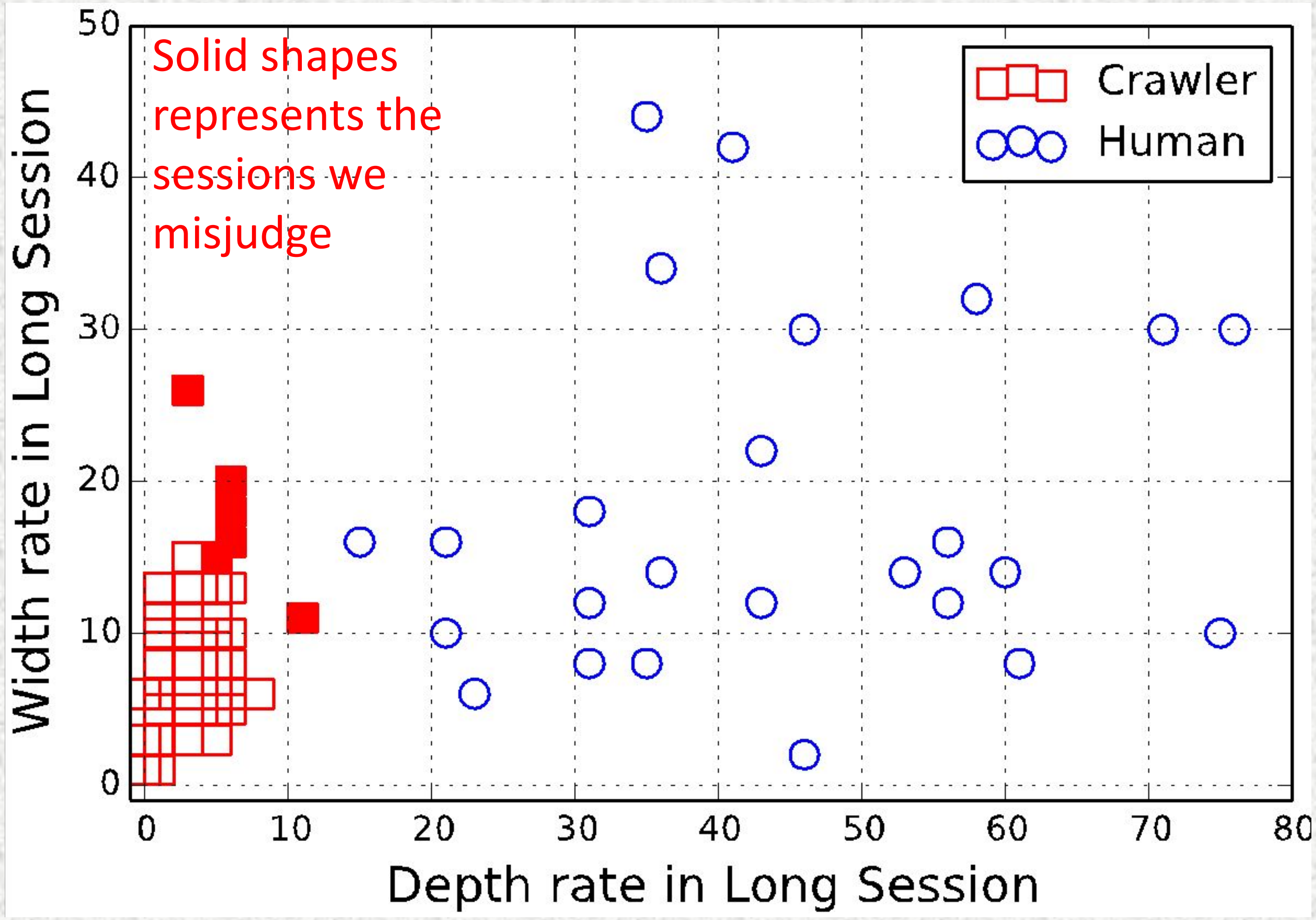
Overhead for Distributed Crawlers

Google Case Study – Heuristic Detection

Visitor IP	URL	Marker
66.249.67.83	home/node/show/12/	home/topic/show/855/;66.249.67.71
66.249.67.77	home/topic/add/	home/home/getmore/13/;66.249.67.83
66.249.67.86	home/policy/	home/user/profile/13/;66.249.67.80
66.249.67.80	home/node/	home/home/getmore/70/;66.249.67.92
66.249.67.71	home/node/show/12/15/	/index.php/node/show/12/8/;66.249.67.77

Example access logs for Detecting Distributed Crawlers

Google Case Study
– machine learning based detection



Depth and Width rate in long session for Google Bots

Outline

- ~~Background~~
- ~~Threat Model~~
- ~~Related Work~~
- ~~Our Solution~~
- ~~System Design~~
- ~~Experiment~~
- Discussion & Limitation
- Conclusion

Usability Issue

- Users could not know the plaintext of URLs
  - Checking titles of pages to identify the content
  - Using bookmark
  - Revealing domain name of every URLs

- Users are only allowed to visit others’ links under a threshold
  - Setting a relatively high threshold
  - For our forum, all normal users have not been classified as crawler because of visiting others’ links

Deployability Issue

- Static web pages
  - Automatically changing all the URLs in scripts

- Dynamic web pages
  - There are different server-side scripting languages
  - It is not possible to design a generic tool for all website servers to adapt their URLs with PathMarker
  - One or two most common functions to generate URLs
    - Integrating markers with these functions

Detection Capability Limitation

- Do not guarantee all crawlers would be captured
    - Accurately mimic human beings' visiting paths

- Still could suppress the efficiency of all crawlers

Future Work

- Crawlers’ path patterns could be classified into three categories

- Baiting Link
  - A kind of link that hardly any normal users would be interested in
  - When a baiting link is visited, a CAPTCHA pops up

- Ensuring crawlers visit the baiting link within limited requests

- How to place the baiting link better?
  - For a Depth-first crawler, it is likely to visit the first link of the next page, which can be where the baiting link located

Outline

- ~~Background~~
- ~~Threat Model~~
- ~~Related Work~~
- ~~Our Solution~~
- ~~System Design~~
- ~~Experiment~~
- ~~Discussion & Limitation~~
- Conclusion

Conclusion

- Anti-crawler system: capturing stealthy persistent crawlers
- Appending URL markers at the end of all URLs
- Calculating accurate path-based features
- Suppressing the crawling efficiency of crawlers who could escape two layers of detections

Any Question?

Thank You!

Shengye Wan

Department of Computer Science

The College of William and Mary