PUBCRAWL: Protecting Users and Businesses from CRAWLers

Grégoire Jacob\textsuperscript{1,3}, Engin Kirda\textsuperscript{2}, Christopher Kruegel\textsuperscript{1}, Giovanni Vigna\textsuperscript{1}

\textsuperscript{1} University of California, Santa Barbara / \textsuperscript{2} Northeastern University / \textsuperscript{3} Télécom SudParis

* image from viralpatel.net

Fri Aug 10 2012
**Introduction:** are crawlers a threat?

**What do web crawlers/spiders do?**
- Browse the web in an automatic and systematic fashion
- Various types:
  - *web indexers* for search engines,
  - *link checkers* for site verification,
  - but also *scrapers* to harvest the content of sites

**When does crawling become an abuse?**
- Unauthorized large-scale crawls over web sites or social networks
- Use the collected data for competing products or services
  *e.g.*, American Airlines-2003, Ryanair-2008, Facebook-2010
- Use the collected data for social engineering or targeted attacks
  *e.g.*, StudiVZ-2009
Introduction: crawler prevention

How can we prevent crawlers from accessing a web site?

- **Robot Exclusion Protocol:**
  - ⊕ Access rules to limit crawlers to certain parts of a site
  - ⊖ Cooperation of the crawler is required

- **User Authentication:**
  - ⊕ Precise tracking of the user activity
  - ⊖ Forcing user to login might not comply with the business model

- **CAPTCHAs and Traps:**
  - ⊕ Tests/Traps that are easily solved/avoided by humans but remain hard for computers
  - ⊖ Potential usability issues and reduced user satisfaction

- **Crawlers need to be identified first to trigger prevention**
Introduction: crawler prevention

Why is the problem of detecting crawlers hard?

- IP-based identification of traffic sources
  - User-agent strings are unreliable
  - A same IP can host multiple users (proxy)
  - Authentication not necessarily available

- Passive detection is constraining
  - Request logs only contains basic information: timing, HTTP header and URL information
  - Request logs contains huge amounts of data to handle
Introduction: crawler prevention

How are crawlers currently detected?

- Learning techniques to extract crawlers’ properties
- HTTP header artifacts:
  - Betraying user-agent, missing referrer, ignored cookies
  - Stealthier crawlers already handle these shortcomings
- Simple traffic statistics:
  - Large request volume, short inter-arrival time, night traffic
  - Large number of users behind a proxy show similar statistics
  - These statistics become inadequate with distributed crawling
- Need robust properties to distinguish:
  - large proxies hosting a large number of users
  - stealthy crawlers mimicking browsers
  - distributed crawlers over multiple sources
Introduction: containing crawlers

Our approach: PUBCRAWL

- Hypotheses on the traffic shape:
  - **User traffic**: high versatility on the short-term, daily regularity.
  - **Crawler traffic**: high stability/versatility on the long-term.
  - **Distributed crawler traffic**: high synchronization across sources.

- Traffic model based on time series
  - **Rational 1**: Independently, time-series can model the traffic shape of a source in a way that is independent of the traffic volume.
  - **Rational 2**: Combined, time-series offer valuable information about the synchronization of sources.

- Machine learning to extract distinctive characteristics of crawler traffic
- Clustering to identify synchronized traffic from crawling campaigns
- Automatic configuration of a containment strategy accordingly
System: PUBCRAWL overview

System architecture

Traffic Logs → HTTP Headers Fields & URLs → Heuristic Detection → Whitelist → Proactive Containment

Traffic Logs → HTTP Headers Fields & URLs → Traffic Shape Detection → Blacklist → Proactive Containment

Traffic Logs → Time Series → Campaign Attribution → Campaigns → Proactive Containment

Incoming Traffic
System: heuristic detection

Detection of suspicious request content

- **Input:**
  - Suspicious values in the HTTP header fields
  - Suspicious visit patterns in the URLs

- **Heuristics:**
  - Original set of heuristics from the state of the art
  - Extended heuristics adapted to social networks
  - Decision by majority vote over heuristics

<table>
<thead>
<tr>
<th>Source</th>
<th>Heuristic</th>
<th>Suspicious check</th>
</tr>
</thead>
<tbody>
<tr>
<td>HTTP</td>
<td>High error rate</td>
<td>404 errors</td>
</tr>
<tr>
<td>HTTP</td>
<td>Suspicious referrer</td>
<td>none or directory query</td>
</tr>
<tr>
<td>HTTP</td>
<td>Unbalanced traffic</td>
<td>99% of traffic coming from a single-user agent</td>
</tr>
<tr>
<td>HTTP</td>
<td>Ignored cookies</td>
<td>new sequence number at every request</td>
</tr>
<tr>
<td>URL</td>
<td>No parameter use</td>
<td>no language choice depending on the profile</td>
</tr>
<tr>
<td>URL</td>
<td>Low page revisit</td>
<td>low overlap in the visited profiles</td>
</tr>
<tr>
<td>URL</td>
<td>Profile sequence</td>
<td>alphabetical order of visited profiles</td>
</tr>
</tbody>
</table>
**System:** traffic shape detection

Detection of crawlers by traffic classification:

- **Input:** time-series with normalized amplitude and common time origin
- **Classification:**
  - Features: Auto-correlation and decomposition analyses
  - Classifiers:
    + Naive Bayes, Association Rules and SVM classifiers
    + Training over user and crawler traffic
    + Decision by majority vote over the classifiers output
System: traffic shape detection

Time series generation

- Counting Process: volume of requests per time interval
- Fundamental differences in traffic shape between users and crawlers

Crawlers: very stable, sudden or no shift

Users: locally noisy, daily regularity, slow shifts
System: traffic shape detection

Auto-correlation Analysis: Sample Auto-Correlation function (SAC)

**Crawlers:** linear decay, strong correlation at small lags, single oscillation, no local spike

**Users:** cut-off decay, multiple oscillation, local spikes at day lags
**System:** traffic shape detection

**Decomposition Analysis:** Trend, Season and Noise components (STL)

**Crawlers:** stable or square trend, predominant trend

**Users:** dispersed trend, predominant season
System: campaign attribution

Detection of crawling campaigns by traffic clustering:

- **Input**: time-series with normalized amplitude and common time origin
- **Clustering**:
  - Similarity: Inverse of the squared Euclidean distance
  - Clustering:
    + Incremental clustering around series medoids
    + Cluster generation controlled by a minimal intra-cluster similarity
    + Candidate clusters selected by medoids of similar amplitude and deviation
**System:** campaign attribution

**Time series synchronization**

- Distributed crawlers from a same campaign are highly synchronized
System: proactive containment

Strategy to trigger active responses:

- *Reduce the impact of active responses over users while limiting the quantity of information possibly leaked by crawlers*

- **Minimal volume sources:**
  - Traffic: user traffic with high probability
  - Minimal amount of information possibly leaked

- **Above average volume sources:**
  - Traffic: traffic coming from crawlers or users behind a proxy
  - Detected crawlers are **blacklisted**
  - Legitimate crawlers and stable proxies are **whitelisted**

- **Low to average volume sources:**
  - Traffic: mixed traffic between users and distributed crawlers
  - Time to collect sufficient traffic to apply classification
  - **Active responses** (e.g., CAPTCHAs) to slow down potential leaks
**System:** proactive containment

**Rational**

- Empirical trade-off between:
  - the acquired knowledge about this source and
  - the potential amount of information it may leak
  - the number of impacted sources
**System:** proactive containment

**Rational**

- Empirical trade-off between:
  - the acquired knowledge about this source and
  - the potential amount of information it may leak
  - the number of impacted sources

![Graph showing the trade-off](image)

- **Green line:** Number of Sources
- **Blue line:** Information potentially leaked
- **Red line:** Knowledge about the Source

**Volume of Requests**
System: proactive containment

Rational

- Empirical trade-off between:
  - the acquired knowledge about this source and
  - the potential amount of information it may leak
  - the number of impacted sources
Evaluation: dataset presentation

Social network traffic logs

- Anonymized traffic from a large social network: 
  public profiles accesses (no authentication required)
- Request logs:
  time, encrypted source IP, encrypted subnet, user-agent, target URL, server response, referrer, cookie
- Filters:
  - obvious crawlers (more than 500,000 requests per day)
  - insufficient traffic (less than 1,000 requests per day)
Evaluation: dataset presentation

Training and testing sets

Traffic Log
1-5 August 2011
(110 million requests)
(10 million IPs)

- Crawler Traffic (704 IPs)
- User Traffic (109 IPs)

Filtered Training Set
(73 million requests)
(813 IPs)

Traffic Log
21-25 August 2011
(102 million requests)
(9 million IPs)

Filtered Testing Set
(62 million requests)
(763 IPs)
Evaluation: training the system

System configuration

- Configuring heuristic thresholds:

<table>
<thead>
<tr>
<th>Accuracy</th>
<th>Former features</th>
<th>New features</th>
<th>Combined features</th>
</tr>
</thead>
<tbody>
<tr>
<td>47.84%</td>
<td>82.50%</td>
<td>77.81%</td>
<td></td>
</tr>
</tbody>
</table>

- Training traffic shape classifiers:

<table>
<thead>
<tr>
<th>Accuracy:</th>
<th>Bayes</th>
<th>Rules</th>
<th>SVM</th>
<th>Vote</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cross validation</td>
<td>98.39%</td>
<td>96.36%</td>
<td>98.55%</td>
<td>98.99%</td>
</tr>
<tr>
<td>Two third split</td>
<td>97.45%</td>
<td>96.19%</td>
<td>95.11%</td>
<td>96.90%</td>
</tr>
</tbody>
</table>

- Tuning the intra-cluster similarity threshold:

<table>
<thead>
<tr>
<th>Precision</th>
<th>Recall</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>99.03%</td>
<td>85.54%</td>
<td>94.35%</td>
</tr>
</tbody>
</table>

- Tuning the proactive containment strategy:

<table>
<thead>
<tr>
<th>Minimal Vol.</th>
<th>Sufficient Vol.</th>
<th>Impacted IPs</th>
<th>Impacted Requests</th>
</tr>
</thead>
<tbody>
<tr>
<td>100 requ.</td>
<td>1000 requ.</td>
<td>0.1%</td>
<td>3.2%</td>
</tr>
</tbody>
</table>
Evaluation: testing the system

System evaluation

- Heuristics detection:

<table>
<thead>
<tr>
<th></th>
<th>Former features</th>
<th>New features</th>
<th>Combined features</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accuracy</td>
<td>38.84%</td>
<td>86.34%</td>
<td>74.19%</td>
</tr>
</tbody>
</table>

- Traffic shape classification:

<table>
<thead>
<tr>
<th></th>
<th>Bayes</th>
<th>Rules</th>
<th>SVM</th>
<th>Vote</th>
</tr>
</thead>
<tbody>
<tr>
<td>Global</td>
<td>93.05%</td>
<td>87.55%</td>
<td>94.36%</td>
<td>94.89%</td>
</tr>
<tr>
<td>Legitimate crawlers</td>
<td>92.54%</td>
<td>87.10%</td>
<td>97.18%</td>
<td>93.95%</td>
</tr>
<tr>
<td>Unauthorized crawlers</td>
<td>88.89%</td>
<td>96.27%</td>
<td>100.00%</td>
<td>100.00%</td>
</tr>
<tr>
<td>Masquerading crawlers</td>
<td>98.27%</td>
<td>86.71%</td>
<td>98.84%</td>
<td>98.84%</td>
</tr>
<tr>
<td>Crawlers (TP/FN)</td>
<td>93.66%</td>
<td>87.68%</td>
<td>97.79%</td>
<td>95.58%</td>
</tr>
<tr>
<td>Users (TN/FP)</td>
<td>82.50%</td>
<td>85.00%</td>
<td>32.50%</td>
<td>82.50%</td>
</tr>
</tbody>
</table>
Evaluation: testing the system

System evaluation

- Campaign attribution:

<table>
<thead>
<tr>
<th>Agent</th>
<th>#Clust.</th>
<th>#ClassC</th>
<th>#IP</th>
<th>Req/day</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Legitimate crawlers</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bingbot</td>
<td>5</td>
<td>11</td>
<td>211</td>
<td>6 million</td>
</tr>
<tr>
<td>Googlebot + Feedfetcher</td>
<td>9</td>
<td>11</td>
<td>113</td>
<td>4 million</td>
</tr>
<tr>
<td>Yahoooslurp</td>
<td>4</td>
<td>9</td>
<td>71</td>
<td>500 thousand</td>
</tr>
<tr>
<td>Baiduspider</td>
<td>1</td>
<td>1</td>
<td>23</td>
<td>50 thousand</td>
</tr>
<tr>
<td>Voilabot</td>
<td>3</td>
<td>3</td>
<td>20</td>
<td>19 thousand</td>
</tr>
<tr>
<td>Facebookexternalhit/1.1</td>
<td>1</td>
<td>1</td>
<td>8</td>
<td>14 thousand</td>
</tr>
<tr>
<td><strong>Crawlers with suspicious agent strings</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;&quot;</td>
<td>2</td>
<td>16</td>
<td>22</td>
<td>330 thousand</td>
</tr>
<tr>
<td>Python-urllib/1.17</td>
<td>2</td>
<td>51</td>
<td>54</td>
<td>140 thousand</td>
</tr>
<tr>
<td>Mozilla(compatible;ICS)</td>
<td>1</td>
<td>10</td>
<td>10</td>
<td>70 thousand</td>
</tr>
<tr>
<td>EventMachine HTTP Client</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td>3 thousand</td>
</tr>
<tr>
<td>Gogospider</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>2 thousand</td>
</tr>
<tr>
<td><strong>Masquerading crawlers</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Gecko/200805906 FireFox</td>
<td>1</td>
<td>10</td>
<td>73</td>
<td>350 thousand</td>
</tr>
<tr>
<td>Gecko/20100101 FireFox</td>
<td>9</td>
<td>12</td>
<td>25</td>
<td>60 thousand</td>
</tr>
<tr>
<td>MSIE6 NT5.2 TencentTraveler</td>
<td>1</td>
<td>1</td>
<td>30</td>
<td>7 thousand</td>
</tr>
<tr>
<td>Mozilla(compatible; Mac OS X)</td>
<td>1</td>
<td>1</td>
<td>4</td>
<td>8 thousand</td>
</tr>
<tr>
<td>googlebot(<a href="mailto:crawl@google.com">crawl@google.com</a>)</td>
<td>1</td>
<td>1</td>
<td>4</td>
<td>1 thousand</td>
</tr>
</tbody>
</table>
### Evaluation: system evasion

#### Evasion techniques and remediation

<table>
<thead>
<tr>
<th>Evasion</th>
<th>Potential remediation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Browser header mimicry</td>
<td>traffic-shape detection</td>
</tr>
<tr>
<td>Traffic shape randomization</td>
<td>traffic-shape detection</td>
</tr>
<tr>
<td>Traffic shape engineering</td>
<td><em>model of user traffic needed</em></td>
</tr>
<tr>
<td>Distributed crawling</td>
<td>campaign attribution</td>
</tr>
<tr>
<td>Traffic de-synchronization</td>
<td><em>robust similarity measure</em></td>
</tr>
</tbody>
</table>
Conclusion: PUBCRAWL

Contributions

- Solution to the detection and prevention of crawlers
- Traffic model relying on time-series:
  - More robust than traditional HTTP field values
  - More robust than simpler statistics over the traffic volume and speed
- Detection based on learning techniques over traffic shape features
- Identification of crawling campaigns by clustering of distributed traffic
- Optimization of the containment strategy according to detection
- Large scale deployment in production at a well-known social network