Spam ain’t as Diverse as It Seems: Throttling OSN Spam with Templates Underneath

Hongyu Gao, Yi Yang, Kai Bu, Yan Chen, Doug Downey, Kathy Lee, Alok Choudhary

Northwestern University, USA
Zhejiang University, China
Background

Among world’s most visited websites by Alexa

http://afrodigit.com/visited-websites-world/

2
1.35 billion monthly active users by Jul 2014

10
284 million users by Oct 2014

14
332 million users by Nov 2014
Background

Users

Interfaces

Spam Filter

OSN Service Provider

Storage
Scary Twitter spam stats

• 2011. **3.5 billion** tweets posted to Twitter every day are spam
  [http://tinyurl.com/p8mqqvs](http://tinyurl.com/p8mqqvs)

• 2014. **14 percent** of Twitter’s user base is bots and spam bots
  [http://tinyurl.com/l755bvm](http://tinyurl.com/l755bvm)
First study to offline detecting and characterizing Social Spam Campaigns (SIGCOMM IMC 2010)
- Largest scale experiment on Facebook then
  - 3.5M user profiles, 187M wall posts
- Confirm spam campaigns in the wild.
  - 200K spam wall posts in 19 significant campaigns.
- Featured in Wall Street Journal, MIT Technology Review and ACM Tech News

Online spam campaign discovery (NDSS 2012)
- Mostly use non-semantics information, syntactic clustering
How Are the Spam Tweets Generated?

Measuring Trend of Twitter Spam
- Download tweets containing popular hashtags
- Visit Twitter retrospectively to identify suspended accounts

- 2011 Twitter data:
  - 17 Million tweets
  - 558,706 spam tweets (>3%)
Template Model

- A macro sequence \((m_1, m_2, \ldots, m_k)\)
- Each macro instantiates differently during spam generation

<table>
<thead>
<tr>
<th>Macro(_1)</th>
<th>Macro(_2)</th>
<th>Macro(_3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Beppe Signori</td>
<td>making out with another man - URL</td>
<td></td>
</tr>
<tr>
<td>Jason Isaacs</td>
<td>making out with another man - URL</td>
<td></td>
</tr>
<tr>
<td>Beppe Signori</td>
<td>is really gay, look at this video URL</td>
<td></td>
</tr>
<tr>
<td>Jason Isaacs</td>
<td>is really gay, look at this video URL</td>
<td></td>
</tr>
<tr>
<td>RIP Jonas Bevacqua</td>
<td>is really gay, look at this video URL</td>
<td></td>
</tr>
</tbody>
</table>

Template = *celebrity names + actions + URL*
The majority of spam is generated with underlying templates.

We collect a smaller 2012 Twitter data containing 46,891 spam tweets.

The prevalence of template-based spam is persistent.

Syntactic only detection is not sufficient!
Semantics Based Spam Detection

- Extract spam template in real time
- Fight spam with its own template
- Detect multiple spam templates simultaneously
Challenges

• Absence of invariant substring in template
  – Prior study assumes the existence of invariant substrings. [Pitsillidis NDSS’10][Zhang NDSS’14]

• Prevalence of noise
  – Spammers extensively add semantically unrelated noise words into spam messages.

• Spam heterogeneity
  – It is hard to obtain a training set containing spam instantiating a single template in practice.
Solutions

• Absence of invariant substring in template
  – Spam template generation without the need for invariant substring.

• Prevalence of noise
  – Automated noise labeling to identify and exclude noise words from template generation.

• Spam heterogeneity
  – Cluster and refine.
• Real-time detection
• The auxiliary spam filter supplies training spam samples
  – Could use black list or any other spam detection systems
  – Heterogeneous filters to avoid evasion
**Single Campaign Template Generation**

Step 1: Compute a “good” common super-sequence

(Majority-Merge algorithm)

- Beppe Signori making out – URL
- Jason Isaacs making out – URL
- Beppe Signori is really gay URL
- Jason Isaacs is really gay URL
- RIP Jonas Bevacqua is really gay URL

---

Super-sequence

<table>
<thead>
<tr>
<th>Beppe Signori</th>
<th>Jason Isaacs</th>
<th>making out</th>
<th>is really gay</th>
<th>- url</th>
<th>RIP Jonas Bevacqua is really gay</th>
<th>url</th>
</tr>
</thead>
<tbody>
<tr>
<td>Beppe Signori</td>
<td>ε</td>
<td>ε</td>
<td>ε</td>
<td>ε</td>
<td>ε</td>
<td>ε</td>
</tr>
<tr>
<td>ε</td>
<td>Jason Isaacs</td>
<td>ε</td>
<td>ε</td>
<td>ε</td>
<td>ε</td>
<td>ε</td>
</tr>
<tr>
<td>ε</td>
<td>ε</td>
<td>ε</td>
<td>ε</td>
<td>ε</td>
<td>ε</td>
<td>ε</td>
</tr>
<tr>
<td>ε</td>
<td>ε</td>
<td>ε</td>
<td>ε</td>
<td>ε</td>
<td>is really gay</td>
<td>ε</td>
</tr>
<tr>
<td>ε</td>
<td>ε</td>
<td>ε</td>
<td>ε</td>
<td>ε</td>
<td>ε</td>
<td>ε</td>
</tr>
<tr>
<td>ε</td>
<td>ε</td>
<td>ε</td>
<td>ε</td>
<td>ε</td>
<td>ε</td>
<td>ε</td>
</tr>
<tr>
<td>ε</td>
<td>ε</td>
<td>ε</td>
<td>ε</td>
<td>ε</td>
<td>ε</td>
<td>ε</td>
</tr>
<tr>
<td>ε</td>
<td>ε</td>
<td>ε</td>
<td>ε</td>
<td>ε</td>
<td>ε</td>
<td>ε</td>
</tr>
<tr>
<td>ε</td>
<td>ε</td>
<td>ε</td>
<td>ε</td>
<td>ε</td>
<td>ε</td>
<td>ε</td>
</tr>
<tr>
<td>ε</td>
<td>ε</td>
<td>ε</td>
<td>ε</td>
<td>ε</td>
<td>ε</td>
<td>ε</td>
</tr>
<tr>
<td>ε</td>
<td>ε</td>
<td>ε</td>
<td>ε</td>
<td>ε</td>
<td>RIP Jonas Bevacqua is really gay</td>
<td>url</td>
</tr>
<tr>
<td>ε</td>
<td>ε</td>
<td>ε</td>
<td>ε</td>
<td>ε</td>
<td>RIP Jonas Bevacqua is really gay</td>
<td>url</td>
</tr>
</tbody>
</table>
### Single Campaign Template Generation

#### Step 2: Matrix columns reduction

<table>
<thead>
<tr>
<th>Beppe Signori</th>
<th>Jason Isaacs making out</th>
<th>is really gay-url</th>
<th>RIP Jonas Bevacqua is really gay-url</th>
</tr>
</thead>
<tbody>
<tr>
<td>Beppe Signori</td>
<td>ε ε making out</td>
<td>is really gay</td>
<td>RIP Jonas Bevacqua is really gay</td>
</tr>
<tr>
<td>ε ε Jason Isaacs making out</td>
<td>is really gay-url</td>
<td>ε ε ε ε ε ε ε ε</td>
<td>RIP Jonas Bevacqua is really gay-url</td>
</tr>
<tr>
<td>Beppe Signori</td>
<td>ε ε ε ε ε ε ε ε ε ε ε ε ε ε</td>
<td>is really gay-url</td>
<td>RIP Jonas Bevacqua is really gay-url</td>
</tr>
<tr>
<td>ε ε ε Jason Isaacs ε ε ε ε ε ε ε ε</td>
<td>is really gay-url</td>
<td>ε ε ε ε ε ε ε ε</td>
<td>RIP Jonas Bevacqua is really gay-url</td>
</tr>
</tbody>
</table>

**Super-sequence**

\[(\text{Beppe}|\varepsilon) (\text{Signori}|\varepsilon) (\text{Jason}|\varepsilon) (\text{Isaacs}|\varepsilon) \ldots\]
## Single Campaign Template Generation

### Step 3: Matrix columns concatenation

<table>
<thead>
<tr>
<th>Beppe Signori</th>
<th>Jason Isaacs</th>
<th>making out -</th>
<th>RIP Jonas Bevacqua</th>
<th>is really gay</th>
<th>url</th>
</tr>
</thead>
<tbody>
<tr>
<td>Beppe Signori</td>
<td>ε ε making</td>
<td>ε ε ε ε ε ε ε ε</td>
<td>RIP Jonas Bevacqua is really gay</td>
<td>url</td>
<td></td>
</tr>
<tr>
<td>Beppe Signori</td>
<td>ε ε Jason</td>
<td>Isaacs making</td>
<td>out - ε ε ε ε</td>
<td>is really gay</td>
<td>url</td>
</tr>
<tr>
<td>Beppe Signori</td>
<td>ε ε ε Jason</td>
<td>Isaacs ε ε ε ε</td>
<td>ε ε ε ε ε</td>
<td>is really gay</td>
<td>url</td>
</tr>
<tr>
<td>ε ε ε Beppe</td>
<td>Signori ε ε</td>
<td>ε ε ε ε ε ε ε ε</td>
<td>ε ε ε ε ε</td>
<td>RIP Jonas Bevacqua is really gay</td>
<td>url</td>
</tr>
<tr>
<td>ε ε ε ε ε ε ε ε Beppe</td>
<td>Signori ε ε</td>
<td>ε ε ε ε ε ε ε ε</td>
<td>ε ε ε ε ε</td>
<td>RIP Jonas Bevacqua is really gay</td>
<td>url</td>
</tr>
</tbody>
</table>

#### Regular Expression Template

<table>
<thead>
<tr>
<th>Beppe Signori</th>
<th>Jason Isaacs</th>
<th>RIP Jonas Bevacqua</th>
<th>is really gay</th>
<th>making out -</th>
<th>url</th>
</tr>
</thead>
<tbody>
<tr>
<td>Beppe Signori</td>
<td>making out -</td>
<td>url</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Jason Isaacs</td>
<td>making out -</td>
<td>url</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Beppe Signori</td>
<td>is really gay</td>
<td>url</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Jason Isaacs</td>
<td>is really gay</td>
<td>url</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RIP Jonas Bevacqua</td>
<td>is really gay</td>
<td>url</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Solutions

• Spam template generation without the need for invariant substring.

• Automated noise labeling to identify and exclude noise words from template generation.

• Cluster and refine for mixture of spam campaigns.
Noise Labeling

Key problem: spammers extensively insert noise words into spam messages
- To draw a larger audience
- To diversify the message

@mentions, #hashtags, popular terms, etc.
Noise Labeling

**Goal:** exclude the noise words from the template generation process.

**Method:** treat noise detection as a sequence labeling task, using Conditional Random Fields (CRFs) approach.

**Output:** a “noise” or “non-noise” label for each word in the message.
Intuition: noise words are popular, but the combination of them are not popular.

Features:

- $freq(t_i)$
- $freq(t_i, t_{i+1})^2 / (freq(t_i)freq(t_{i+1}))$
- $freq(t_{i-1}, t_i)^2 / (freq(t_{i-1})freq(t_i))$

Orthographic features:

- Is capitalized?  
- Is hashtag?
- Is numeric?  
- Is user mention?
Solutions

• Spam template generation without the need for invariant substring.

• Automated noise labeling to identify and exclude noise words from template generation.

• Cluster and refine for mixture of spam campaigns.
Multi-campaign Template Generation

• Problem: in realistic scenario the system observes the mixture of spam instantiating multiple templates, rather than a single one.

• Solution:
  – Part 1, coarse pre-clustering, using standard clustering technique.
  – Part 2, refine the single campaign template generation process, by limiting the ratio of “ε” in the matrix to prune out “outlier” messages.
Recap: Template Generation/Matching Module

- Real-time detection
- The auxiliary spam filter supplies training spam samples
Evaluation Results

• Dataset:
  – 17M tweets generated between June 1, 2011 and July 21, 2011
  – 558,706 spam tweets

• Auxiliary spam filter:
  – The online campaign discovery module (introduced later)
  – 63.3% TP rate, 0.27% FP rate
Detection Accuracy

<table>
<thead>
<tr>
<th>Module</th>
<th>Template Generation</th>
<th>Auxiliary Filter</th>
<th>Combined</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Spam Category</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Template-based</td>
<td>95.7%</td>
<td>70.1%</td>
<td>98.4%</td>
</tr>
<tr>
<td>Paraphrase</td>
<td>51.0%</td>
<td>51.4%</td>
<td>70.1%</td>
</tr>
<tr>
<td>No-content</td>
<td>73.8%</td>
<td>67.0%</td>
<td>83.1%</td>
</tr>
<tr>
<td>Others</td>
<td>18.4%</td>
<td>43.2%</td>
<td>44.7%</td>
</tr>
<tr>
<td><strong>Overall TP</strong></td>
<td>76.2%</td>
<td>63.3%</td>
<td>85.4%</td>
</tr>
<tr>
<td><strong>FP</strong></td>
<td>0.12%</td>
<td>0.27%</td>
<td>0.33%</td>
</tr>
</tbody>
</table>
Top 5 generated templates with the most matching spam:

<table>
<thead>
<tr>
<th>Spam #</th>
<th>Template</th>
</tr>
</thead>
<tbody>
<tr>
<td>11.1%</td>
<td>^ (I wager</td>
</tr>
<tr>
<td>7.2%</td>
<td>^ The ( ε</td>
</tr>
<tr>
<td>6.4%</td>
<td>^ You (will not</td>
</tr>
<tr>
<td>5.0%</td>
<td>^ (Cool</td>
</tr>
<tr>
<td>4.1%</td>
<td>^ You (will not</td>
</tr>
</tbody>
</table>
• Pick the top 5 campaigns
• All campaigns achieve almost 100% detection rate with 0.15% of messages as training samples.
• The system can react to newly emerged campaigns quickly.

Sensitivity for New Campaigns

![Graph showing sensitivity for new campaigns]

- True positive rate (%)
- % of messages used as training samples

Campaigns:
- Campaign No. 1
- Campaign No. 2
- Campaign No. 3
- Campaign No. 4
- Campaign No. 5
The median matching latency grows slowly with template number, less than 8ms.

The largest latency is less than 80ms, unnoticeable to users.
Conclusions

• Tangram: first system to real time extract multiple spam templates without unique invariants.
  – 63% of Twitter spam is generated by templates.
  – Detect 95.7% of template-based spam.
  – Overall TP rate of 85.4% and FP rate of 0.33%.

• Applying text analytics in other security applications
  – Measuring the Description-to-permission Fidelity in Android Applications, CCS 2014
Existing Work, cont’d

• Spam template generation [Pitsillidis NDSS’10][Zhang NDSS’14]
  – How to detect spam without invariant substrings?

• Spammer account detection [Stringhihi ACSAC’10][Yang RAID’11]
  – How to detect spam in real-time?
  – How to detect spam originating from compromised accounts, e.g., in a worm propagation scenario?
Thank you!

http://list.cs.northwestern.edu/

Questions?
Filtering Twitter spam is uniquely challenging

- Twitter exposes developer APIs to make it easy to interact with Twitter platform
- Real-time content is fundamental to Twitter user’s experience

http://tinyurl.com/oxtmmnz