Attention Warfare

John Langford

1. How valuable is your time?

2. How much for “marketing” each year?

“good” vs. “bad” spam

Have they paid for your attention?

1. junk mail

2. junk email

How can we detect junk email?

Signature: The same email to many recipients

Brute force detection

1. Scan all emails

2. Mark the very common emails

Spam Detector Goals

Privacy: Protocol provides no information

Robust: Always detect spam

Distributed: No single point of failure

Efficient: Little space per old email

Incompatible goals

robustness $+$ discrimination $\Rightarrow$ info revealed

Robust enough?

Private enough?

Robust enough

$$\text{Net} = \text{Gross} - \text{Cost}$$

What is the distribution of Gross?

Power law $\Rightarrow$

(x2 sender costs $\Rightarrow$ x$\frac{1}{2}$ spam quantity)

Private enough

email on a public network

$\Rightarrow$ Eve accesses the network elsewhere

Protocol

Based on “tiling” by Andrei Broder

Optimized for:

- distributed computing
- privacy

Two parts:

- hash server
- hash client

Hash server

hash client: “Have you seen this hash before?”

hash server: “248 times”

md5sum

hash any string

$$\{0, 255\}^w \rightarrow \{0, 255\}^{16}$$

Folded md5sum

$$\{0, 255\}^w \rightarrow \{0, 255\}^{16} \rightarrow \{0, 255\}^6$$

Hash client

1. Calculate md5sum of every window

2. sort hashes

3. send largest 3 to random hash server

4. report median value

Analysis

- w = window size

- l = hash length

- k = message size

- m = number of hashes in query

Server Robustness

Denial of service?

Decrease email scope?

Client Robustness

Random hash $\Rightarrow$ unbiased similarity estimate

False hit?

False Hit Analysis

Uniform hash:

$$\forall x \quad \Pr_{s \in \{0,255\}^w}(h(s) = x) = \frac{1}{2^l}$$

Conjecture: md5sum is uniform hash

md5sum uniform $\Rightarrow$ folded md5sum uniform

md5sum uniform $\Rightarrow$

$$\Pr_{s \in \{0,255\}^k}(\text{false hit}_s) \leq \frac{km|S|}{2^l}$$

$|S| = \#$ of previously queried hashes

Problem: english isn't uniform random

Shannon says: 1 bit/byte

$$\Rightarrow w \geq l$$

Client Robustness Summary

Avoidance = sufficiently unique messages

$\Rightarrow$ Canonicalize to preserve meaning

$\Rightarrow$ at least x2 cost (for some scheme $O(n)$ more cost)

Privacy Analysis

md5sum inversion cryptographically hard

smaller $l \Rightarrow$

cryptographic hardness $\Rightarrow$ info theo hardness

Privacy parameters

- $l = 48$ bits
- $w = 100$ bytes
- $m = \min(3, \frac{k}{w})$ hashes

Robustness parameters

- $l = 48$ bits
- $w = 50$ bytes
- $m = 7$ hashes

Optimizations

- Server side time dependence

- Client side time dependence

- “Private” use

Implementation

- Hash server implemented (gs176:11111)
- Hash client implemented (by jslttery)

Current use

Essentially, a human information helper

Easy to setup

/afs/cs/usr/jcl/hasher/README

Conclusion

Simple robust & private protocol to detect spam

Implementation done (try it out)

Future work: integration into spam filters