Spam Mitigation Techniques

Malcolm Robb

mrobb@swin.edu.au

Outline

- Spam 101
- New Techniques
- A spam filtering implementation
- Future work
- Questions?

What is spam?

- A problem is to classify “ham” vs “spam”
- Different definitions include
  - Any unsolicited email
  - Unsolicited Commercial Email (UCE)
  - Any unsolicited bulk email
  - I don't know spam, but I know what I like...
- As users are becoming more sophisticated this is less of a problem

What problems does spam cause?

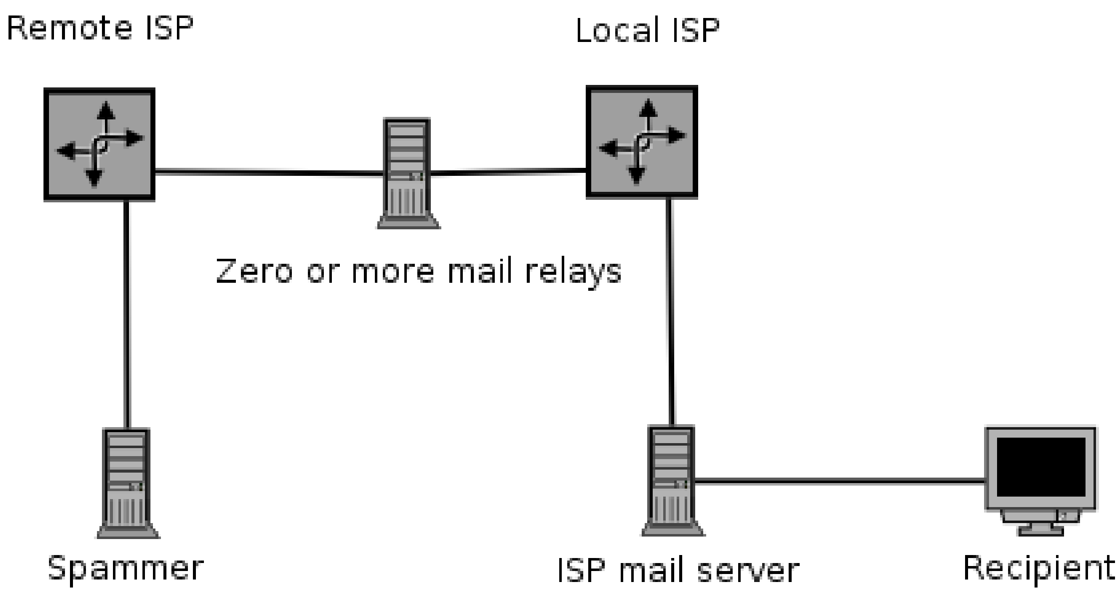
- 60% of total email sent
  - chokes the network
  - reduces the utility of email as a communication medium
- Estimated total worldwide cost of $50 billion in 2005 and volume tends to increase exponentially
- Spam profitability encourages hacking and trade in zombies

How are people fighting spam?

- We concentrate on technical antispam techniques
  - Content-based filtering
    - keyword matching, Bayesian filtering
  - Network level: blacklists and whitelists
    - Traditionally, blacklists are centralised (RBL, ORBS) while whitelists are maintained locally
  - Various other technical (non-automated, non-SMTP) approaches exist
    - Digital sender certificates, Computationally expensive puzzles, or captchas, Sender Policy Framework (SPF)

Where can we combat spam?

- Costs incurred increase closer to the recipient



How well do they work?

- Content based filtering
  - Pros:
    - Accuracy can be very high (99.9%with 0.01%false positives)
    - Tunable for individual recipients
  - Cons
    - Computationally expensive
    - Data is transferred
    - If spam senders are paid for delivery, content filtered spam still counts towards their quotas
    - Spammers constantly develop greater sophistication gaming content filters

Pros and cons (cont'd...)

- Network level filtering in general
  - Pros
    - Inexpensive
    - May reduce bandwidth consumed by spam
    - Can work in tandem with other filtering systems
  - Cons
    - Necessarily incurs nonzero false negative rate

Pros and cons (cont'd...)

- Traditional blacklists
  - Pros
    - Worked successfully to combat open relay spam for years, prior to the “zombie age”
  - Cons
    - Manual administration
    - Slow maintenance, affects
      - recently acquired netblocks
      - secured but previously compromised network
      - erroneous blacklisting and joe jobs
    - Local administrators are subject to agenda of central body

Pros and cons (cont'd...)

- Whitelists
  - Pros
    - Guaranteed delivery from known good senders
  - Cons
    - Difficult to create
    - Some users require unsolicited email, eg from
      - business leads
      - potential employers

Pros and cons (cont'd...)

- Miscellaneous other technology based implementations
  - Pros
    - Attractive solutions exist
    - Easy to speculate
  - Cons
    - Prohibitively expensive to replace existing SMTP infrastructure
    - Computational expense deflated via "SETI @ Home" technology

Effective solutions are difficult to find!

- The infamous "Spam solution form response": http://www.craphound.com/spamsolutions.txt

Your post advocates a
( ) technical ( ) legislative ( ) market-based ( ) vigilante
approach to fighting spam. Your idea will not work. Here is why it won't work.:
( ) Spammers can easily use it to harvest email addresses
( ) Mailing lists and other legitimate email uses would be affected
( ) No one will be able to find the guy or collect the money
( ) It is defenseless against brute force attacks
( ) Users of email will not put up with it
( ) Microsoft will not put up with it
( ) The police will not put up with it
etc...

Novel spam mitigation techniques

- This section is based on the work of Minh Tranh and Grenville Armitage
- Statistical TCP SYN rejection
  - Suggested by Fred Baker in May 2005
  - Track sender reputation
- Rehabilitating blacklists
  - Automate blacklist maintenance
- Poisoning the neighborhood
  - Anticipate future sources of spam

Random Early Detection

- RED (RFC 2309) attempts to anticipate impending congestion in a router queue
  - Congestion probability inferred from average queue size over a duration
  - Congestion is avoided by preemptively dropping packets according to congestion probability
- We apply RED-like concepts to the statistical rejection of inbound TCP connections

Applying RED to email

- Replace queue length with sender reputation
  - = the instantaneous probability of email being rejected by site filters
- If sender's spam probability is less than some predefined threshold, all emails will be passed
  - Analogous to RED's min_{th}
- If sender's spam probability exceeds a maximum threshhold all email is dropped
  - RED's max_{th}

SWINBURNE UNIVERSITY OF TECHNOLOGY

Rehabilitation

- Sender's reputation is gradually improved if spam is not seen
  - Extension: Sending “ham” may more rapidly improve a sender's reputation
- Rehabilitation is intrinsic to the RED scheme
- Useful in the automatic maintenance of blacklists
- Requires current knowledge of offenders so rehabilitation does not proceed

SWINBURNE UNIVERSITY OF TECHNOLOGY

Rehabilitation (cont'd...)

- How long should the rehabilitation interval be?
- Some guiding figures
  - Average Sendmail retry interval
    - Research indicates that this is around one hour
  - Users are typically warned after mail has been undeliverable for a day
- Administrators may set this interval according to their notion of maximum utility

SWINBURNE UNIVERSITY OF TECHNOLOGY

Neighbourhood poisoning

- Anticipate that spammers may be close together in IP space
- Reasonable because addresses are allocated in blocks
- Numerically similar addresses often share administrative control and may be related in other ways
  - Home broadband connections or office networks
  - Hijacked or purchased IP space

SWINBURNE UNIVERSITY OF TECHNOLOGY

Implementation

- Funded by auDA foundation grant, December 2006 – Feb 2007
- Deliverables:
  - Transparent mail server front end
  - Probabilistic rejection
  - Neighbourhood poisoning
  - FreeBSD platform

Implementation Topics

- Design decisions
- Architecture
- Filtering performance
- Configurability

Design decisions

- Packet filtering – Firewall rules vs divert sockets
  - Divert socket
    - Used by FreeBSD's userspace natd
      - reputation for slow performance
        - context switches
        - libalias inefficiency
    - More natural approach to the problem
    - Allows filter to observe connection attempts
    - Allows implementations of packet filtering / manipulation in userspace
    - Portable to Linux (with a divert socket kernel patch)
    - Failure of a divert socket application can result in all port traffic being diverted to the bit bucket!

Design decisions (cont'd...)

- Firewall rule
  - Duplication of state between the blacklist server and the firewall ruleset
  - Firewall implementations may not be optimised for massive rulesets
  - Firewall genericity may make large ruleset optimisation difficult
  - Primitives for state synchronisation between firewall and blacklist server are often limited and nonperformant
  - Complex implementation
  - Reducation of information available to the HRM limits functionality

Design decisions (cont'd...)

- IPFW's lookup tables
  - Are easy to maintain from userspace
  - support extremely large (4mil+) IP address match lists
  - Offer excellent (O(log(N)) average performance via radix trees
  - Were a great relief to find
  - ... but support only IPv4

Design decisions (cont'd...)

- auDA implementation supports both schemes
- Design is flexible enough to support additional firewall implementations

Design decisions (cont'd...)

- Language
  - For Unix system programming, C is the natural choice
  - C++: speed, elegance (?), functionality
  - Most code lies within the OS interface; no great gains from implementation in a higher level language

Design decisions (cont'd...)

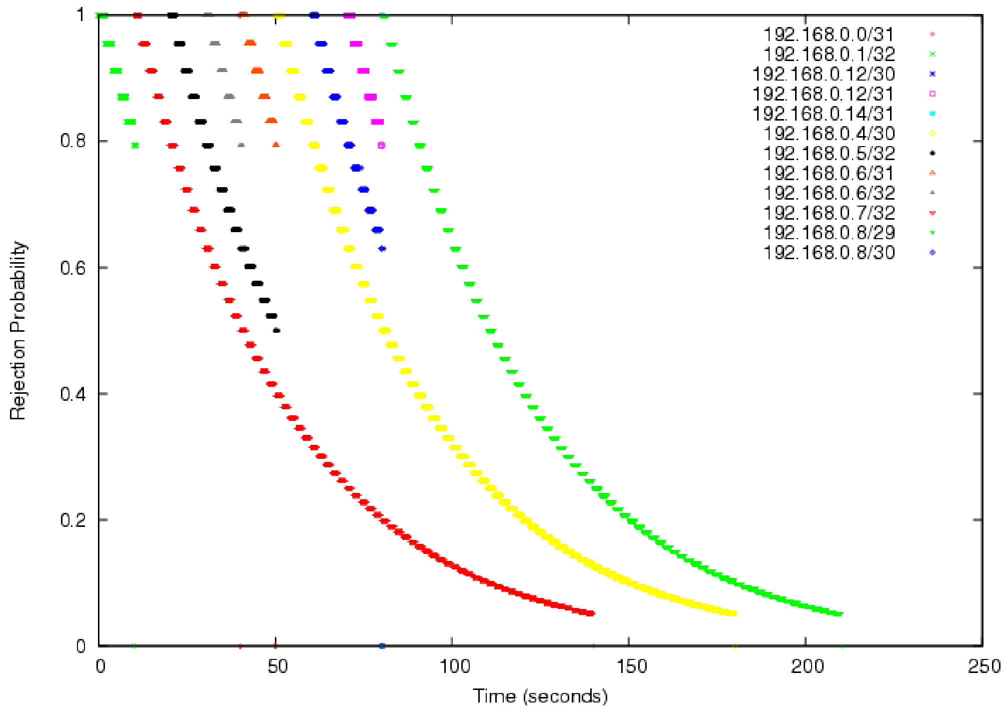
- Communication with email classification engines
  - Text based TCP socket protocol
  - Spam classification engines need not be local to the blacklist server
  - Load testing revealed that this design could be a source of performance problems on a busy server

Design decisions (cont'd...)

- Rehabilitation efficiency
  - Rehabilitation is deferred and heartbeat based
    - Calculations occur when a blacklist entry is accessed
    - A “heartbeat” thread also runs to periodically rehabilitate a subset of blacklist entries
    - The heartbeat thread also performs consolidation and removal of rehabilitated entries
    - When using the rule based filter, entries are never “accessed”. All rehabilitation is performed by the heartbeat in this case.

Design decisions (cont'd...)

- Plotting rehabilitation



Design decisions (cont'd...)

- Plotting rehabilition: the story

t=0: 192.168.0.1/32 (green x) is registered with metric of 1

t=10: 192.168.0.0/31 (red cross) is registered.
192.168.0.1/32 entry subsumed and removed
(denoted by the green dot on the time axis).

t=20: 192.168.0.5/32 (black) added
192.168.0.0 continues to rehabilitate.

t=30 192.168.0.6/32 (grey) added

t=40: 192.168.0.7/32 added (red inverted diamond)

t=40: this is immediately aggregated with 192.168.0.6/32 to form
192.168.0.6/31 (orange triangle).

t=50: 192.168.0.4/30 (yellow square) is added, subsuming 192.168.0.6/31

t=60: 192.168.0.8/30 (blue diamond) added

t=70: 192.168.0.12/31 (purple box) added

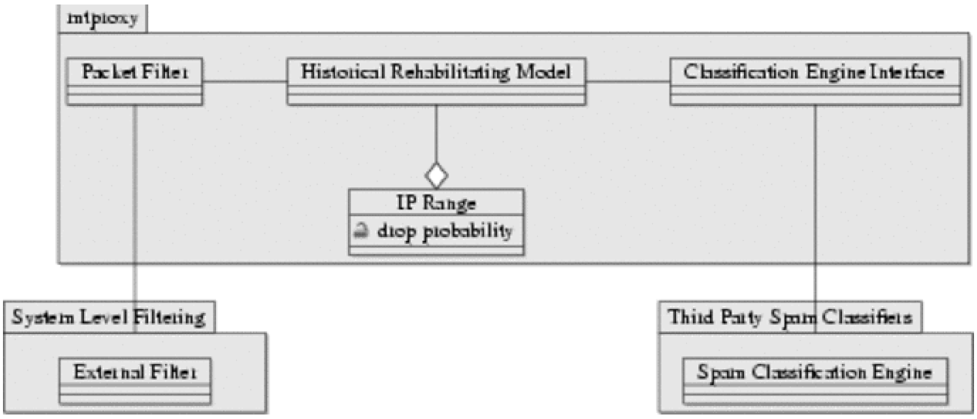
t=80: 192.168.0.14/31 added
192.168.0.12/31, 192.168.0.14/31 aggregated to 192.168.0.12/30
192.168.0.8/30, 192.168.0.12/30 aggregated to 192.168.0.8/29

Design decisions (cont'd...)

- Continuous drop
  - A period of continuous dropping is required once a sender is rejected
  - Without this, a spammer may continue “rolling the dice” rapidly until a connection attempt succeeds
  - May be implemented via ipfw keep-state rule actions

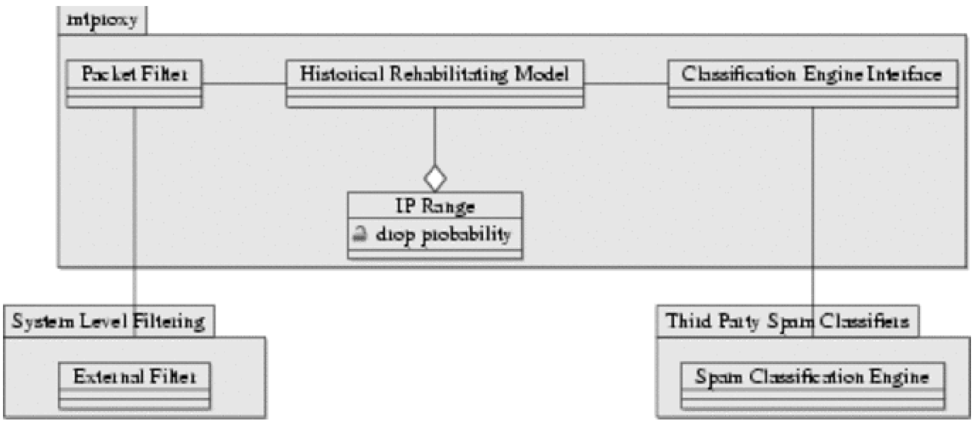
Architecture

- Packet Filter
  - Responsible for dropping or resetting connections
  - Generally the least portable system component
  - Packet filter capabilities affect the requirements of other system components



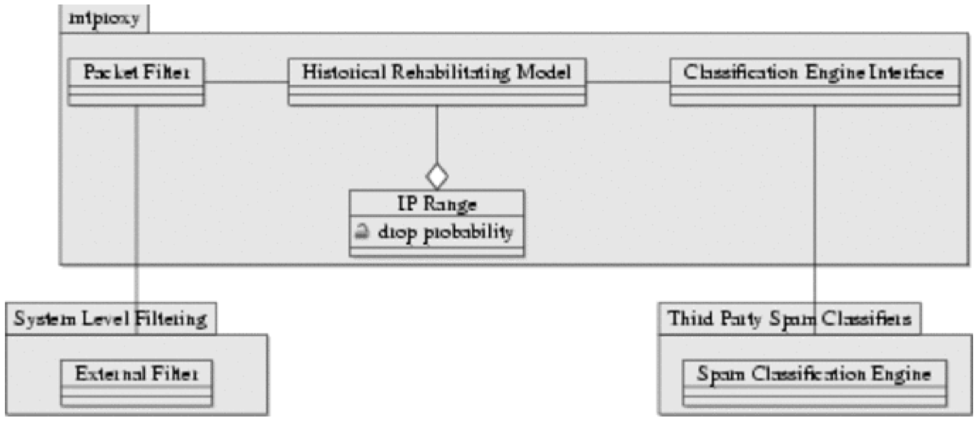
Architecture (cont'd...)

- Historical Rehabilitating Model (HRM)
  - Tracks sender reputation of IP ranges and supervises their rehabilitation
  - Supports both query and observer (push) interfaces for various packet filter clients



Architecture (cont'd...)

- Classification Engine Interface
  - Spam registered via human readable TCP protocol
  - Spam Classification Engines may be “hooked” to register spammers
  - C and Perl libraries and a small executable available to access this interface



Filtering performance

- Comparison of IPFW vs divert sockets
- Not intended to measure production performance!
- Objective of test to place as much load on the blacklist server as possible and measure its response
- CPU idle percentage was chosen as a reasonable performance metric for these tests

Filtering performance (cont'd...)

- Process:
  - Two PCS
    - One with massively multihomed ethernet
      - aliased as 192.168.0.1 – 192.168.255.254/16

```
# time ./aliasif add fxp0 192.168.0.0/16
    0.225u 639.960s 10:43.19 99.5%  31+255k 0+0io 0pf+0w
~> ifconfig fxp0 | wc -l
   65545
```

  - Running a multithreaded data source process, opening multiple TCP connections and pumping 40kb of data through each
  - 2.5 GHz Celeron, 1Gb RAM

Filtering performance (cont'd...)

- Blacklist server test host
  - Running blacklist server in various configurations
  - Mail server emulation by a data sink process, accepting multiple simultaneous connections and discarding data
- Intel PIII 800 MHz, 128 Mb RAM

Filtering performance (cont'd...)

- The Showdown...
  - Four tests

Test	A	B	C	D	E
Blacklist server	none	Divert socket listener, no divert rule	Divert socket listener, setup traffic only	Divert socket listener, all port traffic	IPFW rule based filter
Spam registrations	none	25%	25%	25%	25%
CPU Idle %	63%	52.9%	47.7%	33.8%	47.5%

Configurability

- Configurable runtime userid, in case the classification client engine is compromised.
  - Privileges required to update firewall and to clean up on process exit.
  - Isolate privileged code in forked process
- Filtering protocol is selectable via the configuration file.
  - Probabilistic rejection emulated with use of multiple rules
- IPFW keep-state may be deactivated if desired

Configurability (cont'd...)

- Rehabilitation parameters
  - Minimum and maximum meaningful drop probabilities
    - configure the RED min_{th} and max_{th} values.
  - Probability of rejecting an open TCP session on packet reception, relative to rejecting a connection attempt.
    - Currently all open connections are dropped when connection rejection probability exceeds max_{th}.
  - Decay factor controlling how quickly the reputation of a sender is rehabilitated.

Configurability (cont'd...)

- Rejection protocol
  - TCP reset
  - ICMP unreachable
  - Silent drop

Future work

- Different rehabilitating models
- Multiple firewall rule based filters
- Blacklist state persistence (?)
- Exploit extra information offered by divert sockets

Questions?

- How does a sender's reject probability change on receipt of spam?
  - This can vary with the implementation.
  - Our current model adopts the higher of the current reject probability of the sender, and that assigned to the incoming mail.
  - Another option is to adopt a more RED-like approach, selecting a value between the existing and new reject probabilities.

Questions (cont'd...)

- Where can this system be deployed?
  - We anticipate deployment on mail server systems, but the system is flexible enough to be deployed in other positions upstream.

Questions (cont'd...)

- Have we performed analysis that might indicate an optimal value for the rehabilitation rate?
  - A value in the order of seconds will offer some protection against spam flooding
  - A 30 minute rehabilitation period should interoperate with sendmail retry times to minimise delays in the case of an infrequently spammy source.

Questions (cont'd...)

- A rehabilitation period closer to a day might be more suitable for those less tolerant of occasional spam.
- "Optimal" in this case is relative to the philosophy of individual mail server administrators.

SWINBURNE UNIVERSITY OF TECHNOLOGY

Questions (cont'd...)

- How will the implementation be made available?
  - Under an open source / free software license yet to be determined.
  - Available for download via the CAIA website, and potentially SourceForge later down the track.

SWINBURNE UNIVERSITY OF TECHNOLOGY