SoYoung Park
11/08/01

Paper Topic: Privacy and Censorship resistance

The design, implementation and operation of an email pseudonym server (Sorry, this is a bit long)

important points

- nym.alias.net: a pseudonym email system that was designed with goals that allowed real users outside of the research community, provided secure secrecy of the users' identities, and provided a robust email service.
- Authors first describe the system, then distill from their experience of operating nym.alias.net methods of service abuse and ways to counteract them.

  harassement: a nym user anonymously sends offensive or harassing email to unwanting recipients
  solution: recipient requests server to block any message from that nym to the recipient.

  exponential mail loop: nym's reply block points back to itself
  solution: limit the number of mail a nym can receive each day; disable account and notify nym owner when the limit is exceeded
  Note: the authors never experienced harassment and exponential mail loop. However, the measures in place for those attacks helped deal with other attacks.

  bulk mailing: attacker sends out mail to many recipients; can fill up mail spool queue
  solution: keep a daily limit of outgoing messages

  mail-bomb: attacker generates a large number of messages to one recipient
  solution: keep a per-sender, per-host maxima on mail deliveries; additional sends result in temporary error codes, thereby delaying large bursts of traffic

  reverse mail-bomb: a mail bomb whose "From" address is forged to the victim's address
  solution: this attack to the `help` address which sends autoreplies were dealt by including the sender's mail header with the response

  encrypted mail-bomb: attacker creates a nym for the victim's address and subscribes that nym to a high-traffic mailing address
  solution: require confirmation of new reply blocks

  create many accounts: create large number of accounts to hog resources that would otherwise go to valid users
  solution: each account needs a reply block, so requiring reply block confirmation slowed the attacker down enough to make this attack unattractive

  spam: unsolicited commercial email
  partial solutions: (1) *nobcc* configuration option - worked except for nym users subscribing to mailing lists. (2) "spam-trap" accounts to identify spam source and delay their mail - didn't work because spam sent through a remailer delays the remailer. (3) refuse mail that the smtpd server couldn't bounce.

- The types of abuse attacks are categorized into three categories: conventional attacks, content-based abuse, and overloading. Conventional attacks include mail bombs and those that exploit vulnerabilities in the server's OS. These are are not particular to anonymous services.
- Content-based abuse is difficult to handle through censorship. Therefore, the mail providers must help recipients avoid unwanted mail, while keeping the senders unaware that their messages are being filtered out.
- Overload attacks can be dealt by applying back pressure to the senders. Two ways are introducing server delays and increasing human effort involved in the attacks.

deficiencies

- Today, spam is probably the most common complaint regarding email service. (Actually it's my only complaint so far.) I was disappointed that the paper did not include an effective solution for this. The nobcc method wouldn't work in the CMU environment because internal spam (some of which I would like to receive) often uses the bcc field. Also, spam are often bounceable.

conclusions

- What is interesting about his study is that the lessons learned in dealing with convensional attacks have more general applications. The ways of dealing with content-based abuse are also applicable to non-anonymous email providers. Many web-based email providers support block of particular sender addresses.

Publius: a robust, tamper-evident, censorship-resistant web publishing system

important points

- The authors motivate the work by saying that Internet is unable to resist censorship. The shown example illustrates legal censorship, not technological such as discussed in the other paper.
- Publius: an anonymous web publishing system that makes censorship of content difficult, hides publisher's identity, enables updating/deleting of published material, and has a web interface for browsing the content.
- Its design goals and how they were met:
  - resistance to censorship: third party cannot easily alter or delete published material without having read/write access to the servers
  - tamper detection: URL of content is cryptographically linked to the actual content and a share of the encryption/decryption key, so altering the content results in a dangling URL
  - source anonymity: client proxy does all the encryption to hide the publisher's identity; published material resides on servers that have no idea who the publisher is
  - updateble material: the publisher can use encrypted passwords to send updates/deletions to the server
  - deniable content: third parties do not know the published content because it is encrypted
  - fault tolerant: multiple servers host the content
  - persistent: published materials do not expire
  - extensible: connection-based anonymity tools can further protect user identity
  - freely available: open source code

deficiencies

- Publius supports only static web content, javascript and limited java applets.
- Updateble published content is vulnerable to URL redirection attacks.
- Retrival of a web page and its linked pages require decryption, which may be expensive.
- Choosing the minimum k shares of the key needed to decrypt the content and the number of copies of the content to retrieve is not obvious in view of possible attackers. k must not be less than the number of shares the attacker has corrupted. The number of copies retrieved also must not be less than the number the attacker as corrupted. But retrieving more copies would incur significant processing and network latency.

conclusions about system building

- The ability to update and delete published material are wonderful tools for the publisher. But they also provide the attackers more ways to damage the published content. Adding more features means adding more protective measures, and the protective measures should be a part of the design from the beginning.