Robert J. Hall

How to Avoid Unwanted Email

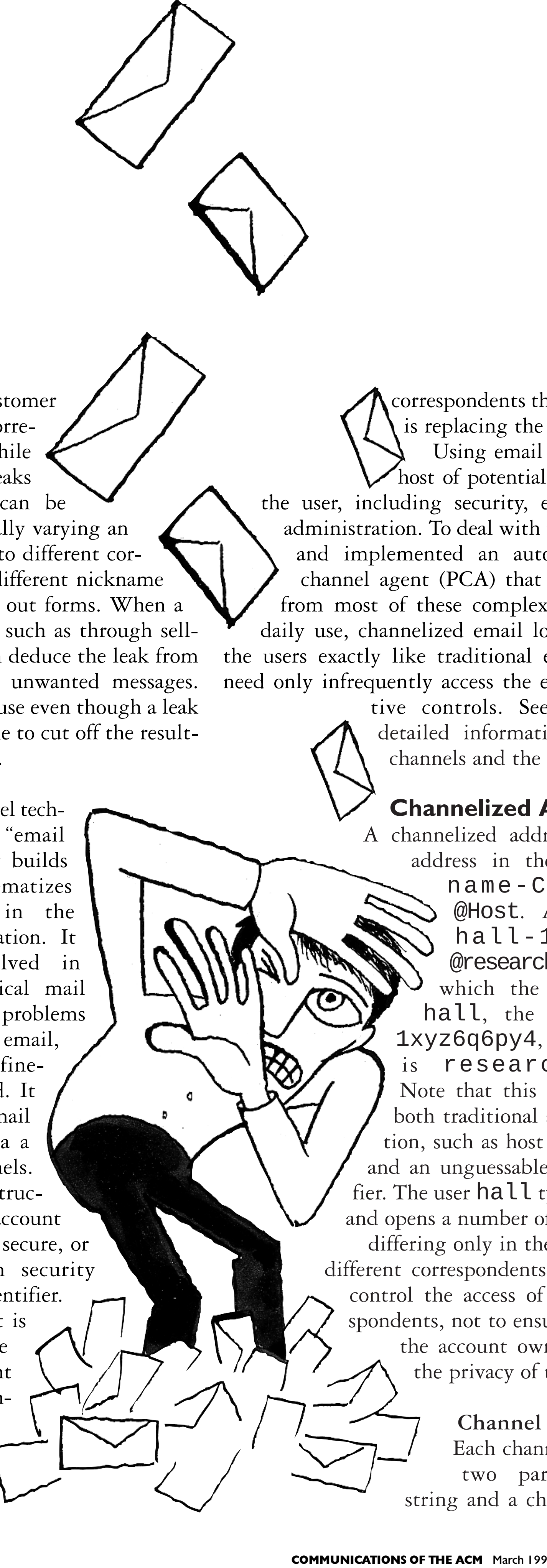
Despite looking like conventional email, a channelized email address and its related agent allow email users to reliably cut off unwanted correspondents.

Unwanted communication ranges from nuisance (junk mail) to annoyance (telemarketing) to dangerous to the very medium conveying the message (junk fax, obscene or harassing telephone calls). The usefulness of email is seriously threatened by the commercialization of the Internet because it is easier than ever to collect address lists and cheaper than ever to mass-distribute messages.

If companies spent as much money sending junk email as they do sending junk physical mail, an established Internet user would likely get more than 100 junk messages per day. Every time a user sends a message to a public newsgroup or list, fills out a Web form, or mails in a product registration card, the server cheaply obtains an email address and usually some indication of the user's interests.

This information is then sold to marketing firms that easily automate mass emailings of advertisements, surveys, and other annoyances that cost the user connect time and, worse, valuable attention. More sinister unwanted email is becoming common as well, including harassing and hate mail.

The main technique today for avoiding unwanted communication is to restrict the set of people to whom users give their addresses. For example, people pay to avoid having their phone numbers listed; in email, people sometimes maintain multiple email accounts, using different accounts for different purposes, such as commercial vs. personal. This unlisted address approach is expensive and slow to recover from security breaches; if an address is leaked to an adversary, the only alternative is to pay the service provider to change it (often a lengthy process). Once



the address is changed, the customer has to notify all legitimate correspondents of the change while keeping it from adversaries. Leaks of physical mail addresses can be crudely located by systematically varying an address slightly as it is given to different correspondents, by, say, using a different nickname or middle initial when filling out forms. When a correspondent leaks a variant, such as through selling a mailing list, the user can deduce the leak from the address used on ensuing unwanted messages. This technique is limited because even though a leak can be traced, little can be done to cut off the resulting unwanted communication.

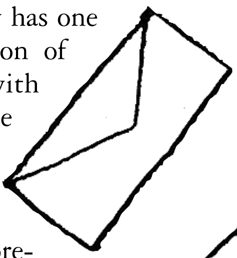
Here I offer a novel technique called "email channels" that builds on and systematizes these ideas in the domain of email communication. It solves the problems involved in unlisted numbers and physical mail addresses, as well as additional problems introduced by the nature of email, providing a light-weight, fine-grained access-control method. It works like this: A user's email account is made accessible via a user-controlled set of channels. Each channel has a distinct structured address containing the account name and a cryptographically secure, or unguessable, pseudo-random security string known as a channel identifier. Each legitimate correspondent is allowed to know one of these access addresses. The account owner is provided simple controls for opening a new channel, closing a channel, and switching a channel by notifying selected correspondents that a new channel is replacing the current one.

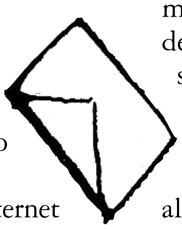
Using email channels raises a host of potential complexities for the user, including security, ease of use, and administration. To deal with them, I designed and implemented an automated personal channel agent (PCA) that shields the user from most of these complexities. In routine daily use, channelized email looks and feels to the users exactly like traditional email, and users need only infrequently access the extra administrative controls. See [6] for more detailed information about email channels and the PCA.

Channelized Addresses

A channelized address is an email address in the form `User-name-ChannelID-@Host`. An example is `hall-1xyz6q6py4-@research.att.com` in which the user's name is `hall`, the channel ID is `1xyz6q6py4`, and the host is `research.att.com`. Note that this address contains both traditional address information, such as host and user names, and an unguessable channel identifier. The user `hall` typically allocates and opens a number of these addresses, differing only in the channel ID, for different correspondents. The goal is to control the access of potential correspondents, not to ensure anonymity of the account owner or guarantee the privacy of the messages.

Channel identifiers. Each channel identifier has two parts: a security string and a channel class indicator. It is critical that channel identifiers be practically unguessable, even when an adversary knows several of the user's other channel identifiers. Thus, the prototype generates security strings pseudo-randomly using the cryptographically secure Blum-BlumShub (BBS) generator [1], with a modulus size larger than 1,024 bits. See Schneier [11] for other candidate generators. A channel ID contains 45 pseudo-random bits. This large number of bits implies that if a user maintains 128 open channels, an adversary has one chance in about 275 billion of guessing an open channel with one message. A brute force attack, sending more than 100 billion messages to the same host, is impractical in today's Internet. Moreover, the security of BBS [1] implies that adversaries who know previously generated bits have essentially no advantage in guessing further bits.





Due to character-set restrictions in Internet mail protocols, the 45 pseudo-random bits are encoded into strictly alphanumeric ASCII characters five bits at a time, using only one case of the alphabet and the digits 3 through 8. I use this base-32 scheme rather than the more standard base-64 encoding, because the latter uses both cases of the alphabet, and not all mail systems on the Internet maintain the alphabetic case of header fields. When a message is received, alphabetic case is ignored in comparing the channel ID to those in active channels.

The channel class indicates how mail on that channel can be treated by the recipient. The current prototype implements three classes:

- Class 0, which indicates a send-only channel, t hat is, one that is permanently closed. These channels are useful as return addresses when a user wants to send a message to a public or adversarial address without giving away any access at all.
- Class 1, which indicates a private channel. The user expects mail from a known set of correspondents on such a channel. Mail from other correspondents may be ignored on it.
- Class 2, which indicates a public channel. Previously unknown correspondents may send on such a channel.

In the future, I plan to implement a richer class scheme, including the following classes:

0. SendOnly
1. Private
2. Permanent Public
3. Temporary Public
4. Commercial
9. Introductory

Thus, a channel identifier has the form `Cxxxxxxxxx`, where `C` is a digit indicating the class and the `x`s encode the security string.

Applications. The multiple channels idea has several applications. For example, how can a user participate in a public forum, such as a mailing list, without giving away access? At subscription time, the user sends a public channel address to the list maintainer. All messages sent to the list will be delivered to the user on this channel. However, to send a message to the list itself, the user uses a send-only return address. Anyone wishing to reply has to send to the entire list.

Users wishing to allow private replies can allocate a limited-lifetime public channel, using it as the return address, perhaps explicitly indicating when it is to be deactivated. Users wishing to respond to the post can do so privately for a short while, but firms collecting interest-based mailing lists are left with closed channels after the time-out period. Users can always choose to upgrade a correspondent to a permanent channel once contact is made.

Channels and list servers together can be used to implement private mailing lists, allowing groups to confer without requiring that they all have direct channels to each other, while prohibiting outsiders from sending to the group. The idea is simply to establish a list server with an unguessable address known only to list members. Note that list members need not have direct channels to each other, so a private mailing list might be useful, for example, when a single buyer needs to have a group discussion, such as an auction, with vendors that are mutually adversarial.

Channelized email can also enhance the effectiveness of email agents and filters [2, 5] by providing categorization based on which correspondents are presumed to know which channels. For example, when filling out a registration form for a product, one can use a particular public channel. The filter could be instructed to classify all traffic on that channel as lower priority than traffic on more personal channels. Furthermore, once electronic money becomes widely used, one can also implement pay-

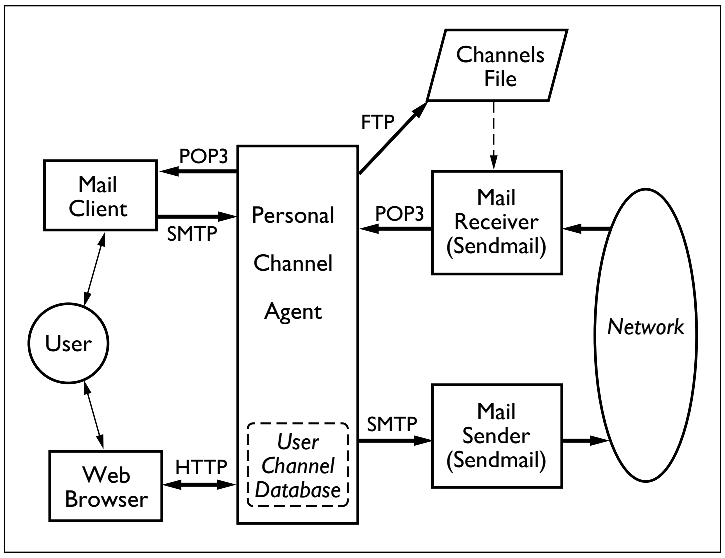


Figure 1. Block diagram of the Personal Channel Agent prototype

per-view channels. The idea is that the channel agent accepts a message on a pay-per-view channel only if accompanied by enough e-money to pay for the user's time viewing the message. Compensating the user for viewing ads, surveys, and more may increase the effectiveness of such marketing tools (as coupons do). On an authenticated channel, the filter rejects messages not digitally signed [4, 10, 11] by an expected correspondent. Note that an authenticated channel could even have a well-known identifier, such as `1AUTHENTIC`, since unauthenticated messages are discarded unseen.

Another useful synergy of channels and email agents is the idea of the introductory channel, or a public, pay-per-view channel with a well-known address. Each channel user with a powerful filtering agent, such as the Andrew system's FLAMES language [2], can establish a well-known public channel identifier, such as `9INTRODUCE`. A message to `user-9INTRODUCE-@host` is automatically handled by first (politely) demanding a reasonable fee (say, $1.00) for reading it, while promising to refund the fee if the message is subsequently determined to be a legitimate attempt at contact and not just junk mail. If the message is junk, the user simply keeps the fee. Such a channel address could be published in directories. The risk of unwanted email is reduced arbitrarily by setting an appropriate access fee for unknown correspondents, since there is presumably a price advertisers will not pay for mass mailings. Yet charging a fee still allows access to long-lost friends and relatives, since the fee is immediately refunded in such cases.

Implementation. It is easy for one's mail server to allow flexibility for channelized addresses. In one prototype (see Figure 1), a modified Unix `sendmail` [3] parses addresses, checking the user part in the system password file as usual and matching the channel ID part against a list of open channels maintained in the user's `channels` file. The message is bounced if either the channel is not open or there is no channel ID present in the address. While this implementation is based on `sendmail`, analogous changes should be straightforward for other mail-processing systems. In a recent all-Java channels implementation, a separate channel bouncer component performed this function before forwarding accepted messages to the regular mail server.

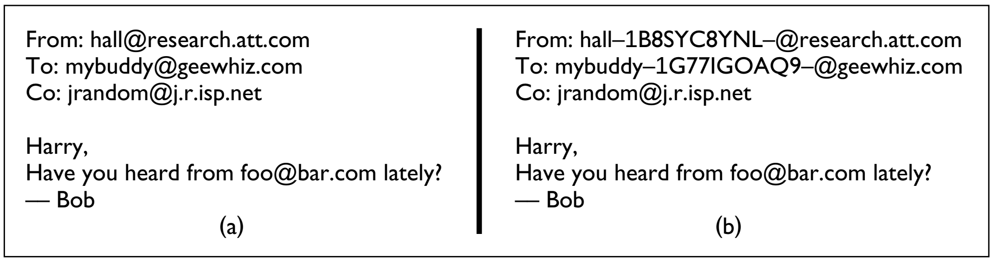
Security. The success of the channels approach requires that the user's mail server, client machine, and the local network connecting them cannot be systematically eavesdropped on by an adversary; otherwise, the eavesdropper would have access to all open channels appearing in the user's mail traffic. While this assumption about server security may not hold in all cases, it is plausible when, for example, the server is run by a reputable commercial online service. In that case, the server and at least part of the network are physically secure and administered competently. Moreover, users connect via modems over traditional voice lines, where eavesdropping requires relatively expensive hardware techniques, unlike connections over ethernet where peer hosts can freely snoop on the packet stream.

Note that the channels approach does not require the entire network to be impervious to eavesdropping. By giving correspondents individual access channels, the user discovers immediately which correspondent has breached security (either accidentally or by being eavesdropped). At that point, the user can either switch the channel, if the breach was a one-time occurrence, or establish a cryptographically authenticated channel. (This authentication feature is not yet implemented in the prototype.)

The Personal Channel Agent

Maintaining multiple channels manually would be cumbersome and error-prone, leading to several problems:

- *Return address.* Remembering which channel to use as your return address for a particular


From: hall@research.att.com
To: mybuddy@geewhiz.com
Co: jrandom@j.r.isp.net

Harry,
Have you heard from foo@bar.com lately?
— Bob

(a)

From: hall–1B8SYC8YNL–@research.att.com
To: mybuddy–1G77IGOAQ9–@geewhiz.com
Co: jrandom@j.r.isp.net

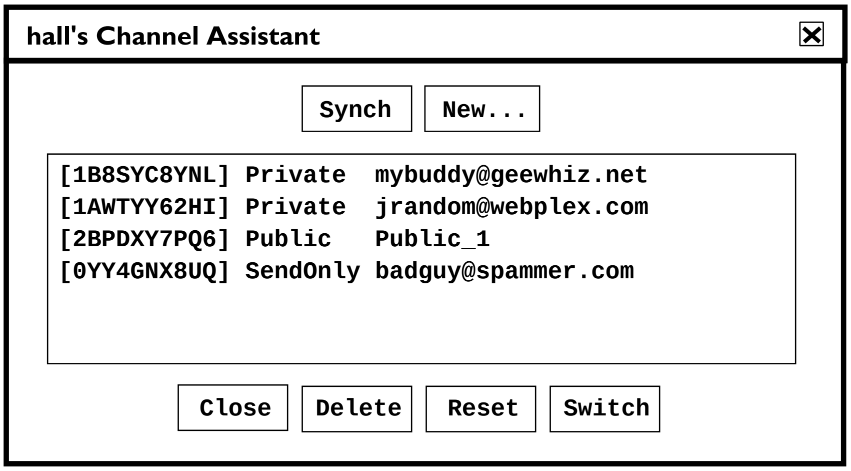
Harry,
Have you heard from foo@bar.com lately?
— Bob

(b)

Figure 2. Address rewriting performed by the PCA; (a) message created by sender; (b) message actually transmitted to geewhiz.com.

user would be onerous.

- *Cc.* When sending to multiple recipients, security is breached if one simply includes everyone's channelized addresses, since it is unlikely that every reader is authorized for each other's channel. For example, suppose one sends a message to a mailing list and cc's a friend's private channel address. The cc is visible to all list readers, so all gain unauthorized access to the friend.
- *Reply/forward.* People frequently include a received message when replying to or forwarding it. If the message contains channel IDs, the user must remember to edit them out to avoid leaks.
- *Anomaly tracking.* It is useful to notice when users send on channels they are not authorized to use, so leaks can be isolated when they become a problem. However, the problem may take a while to appear, as more and more junk traffic builds up on a channel and the original leaks are forgotten.



```
[1B8SYC8YNL] Private  mybuddy@geewhiz.net
[1AWTYY62HI] Private  jrandom@webplex.com
[2BPDXY7PQ6] Public   Public_1
[0YY4GNX8UQ] SendOnly badguy@spammer.com
```

Figure 3. Administrative interface for user Hall's PCA

The PCA I designed and implemented manages these complexities on behalf of the user.

PCA implementation. Figure 1 shows how the PCA prototype fits into an email system. Conceptually, the PCA acts as an email proxy, sitting between the user's mail client and the mail server itself, with a Web browser or desktop window allowing administration of the PCA. All PCA interfaces use standard protocols, such as SMTP [9], POP3 [7], HTTP, and FTP, to interact with clients and servers, so no special client software is needed to use it. This proxy positioning allows the PCA to perform bookkeeping functions autonomously on both incoming and outgoing messages, shielding the user from channel-induced complexities.

This architecture allows the PCA to run on a host separate from the mail server's host, so any additional computational load incurred by the PCA can be distributed. Alternatively, the PCA could run on the same host if desired. The only additional load necessarily incurred by the mail server is in parsing the address (insignificant) plus the time to check the channels file. This additional workload is significant only for large channel files or slow file access. If users want to keep open many channels, the PCA can store channel identifiers in a database format supporting faster access than that available from a flat file.

A key part of the PCA is the user channel database (UCDB) whose primary purpose is to record two mappings: the channel map and the correspondent-address map. The channel map associates each correspondent with the channel on which the user expects to receive mail. The correspondent-address map associates each correspondent's user and host names with the channel ID on which to send to the correspondent, if any. In the current implementation, each correspondent is allowed at most one channel. While it might initially seem desirable to allow multiple channels per correspondent, recall that the primary purpose of the channels mechanism is to deny access by denying knowledge. No security is gained by a single person knowing two or more access channels for a correspondent. Instead, the logical separation of traffic from a single user can be implemented using existing email filtering techniques [2, 5].

Address rewriting. The PCA rewrites the header and envelope information of each message as it comes in or goes out, leaving the body unaltered. For incoming messages, it removes channel IDs from all header addresses before serving the message to the client. Header rewriting solves the reply/forward problem, because the header of the included original contains no channel IDs. Figure 2(a) shows the user's view in the mail client; Figure 2(b) shows what is actually transmitted and received.

For outgoing messages, the PCA puts back channel IDs selectively before forwarding the message to the network. For a single-recipient message, the PCA simply obtains the appropriate to-channel and from-channel to use from the UCDB of the sender and respectively puts them into the recipient and sender fields (in both the message headers and the SMTP envelope). This automatic addition of channel IDs solves the return address problem.

Multi-recipient messages are copied once per recipient listed in the SMTP envelope, and each copy is tailored to that recipient. Tailored copying solves the cc problem, because each recipient receives exactly one copy of the message containing only information s/he already knows.

Thus, to the user, virtually all messages appear without channel IDs, and email looks and feels like traditional email. But why rewrite headers at all? Why not put channel IDs only into the envelope and not insert them into header lines? The primary reasons are for interoperation with non-channel users and with non-SMTP mail systems. A non-channel user expects a valid return address to appear in the `From` field and puts a channelized address in the `To` field, leading to the return address and reply/forward problems. Moreover, some non-SMTP mail systems do not separate the header from envelope information.

Anomaly detection. The PCA checks each incoming message to determine whether the sender is expected to send on the channel the incoming message arrived on. Messages to private channels are checked to see whether the sender is a member of the channel. If not, the user is notified (once for a given user and channel) and the event logged in the UCDB. This notification and logging is not done for public channels, because one expects previously unknown correspondents to send on public channels.

Administrative interface. Users who need to open, close, create, delete, or switch channels use the PCA's administrative interface. One prototype serves this interface as an HTML form via HTTP, allowing the PCA to be on a machine other than the user's client machine if desired, while the Java-based prototype presents it in a window on the user's desktop (see Figure 3). The `Synch` button sends a message to the channels server, synchronizing its representation of the user's list of open channels with that of the PCA. The `New` button initiates a dialog allowing the user to create a new channel. The display list provides a view on the user's channel database, while the buttons below it enable operations on individual entries, such as closing a channel, switching channels, and deleting the entire entry.

Channel switching. It may sometimes be desirable to switch a correspondent from one channel to another, because either the old channel has been leaked to too many adversaries or the user wishes to upgrade the correspondent's access, from, say, public to private or temporary to permanent. If the correspondent does not use a PCA, upgrading requires notifying the user to make a manual address book change. In this case, the PCA helps only in sending out a notification message.

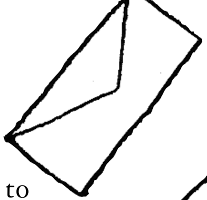
If the correspondent also uses a PCA, the switching can be automated via a channel switching protocol, allowing the user's PCA to make a change in the correspondent-address map of the correspondent's UCDB. However, such a protocol introduces a security risk; for example, an insecure protocol might allow a PCA to be tricked into sending private messages to a public forum.

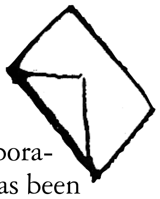
Limitations and Future Work

Limitations of the channels approach involve the following functional categories:

Usability. From the client's viewpoint, a PCA routinely makes channels transparent to the email user. There are, however, several occasions when extra operations must be performed, including:

- When a message is to be sent to a new correspondent, the correspondent's channelized address must be entered in the `To` field of the email client. The PCA then remembers the channel ID and inserts it into succeeding messages to that correspondent.
- The user has to use the administrative interface to allocate new channels, such as for use in mailing lists, as temporary reply channels, and to close and switch channels.
- Malicious actions by an eavesdropper or interdicter may have to be countered by changing the security policy for a correspondent, such as





by switching him/her to an authenticated channel.

I have tested the prototype in a laboratory setting, and initial experience has been positive; for example, the extra operations are relatively infrequent and have seemed easy to perform and understand. However, it is an open question whether most users will come to the same conclusion, particularly when real-world settings are rife with financially motivated adversaries. Future work will test the prototype under more realistic conditions to help settle such questions.

Interoperation with traditional email. While a correspondent who does not use a PCA must directly use the user's channelized address, most mail clients provide online address books, eliminating the need to remember or type the longer address. Due to the cc and reply/forward problems, the channel ID may be leaked when the correspondent sends a multi-recipient message. Automatic channel switching is not possible, but users' PCAs can automatically generate notifications to each correspondent on the changing channel, leaving it to the correspondents to update their own address books.

Directories. Any approach based on not telling everyone how to reach you appears to conflict with directories that tell everyone how to reach you. This tension results from wanting to be reachable by people, yet not wanting to have one's time and attention wasted. Channels help resolve this tension in two ways: One is more willing to publish an easily changeable address than a permanent address, and when e-money is commonplace, users can publish introductory channels in directories, allowing access to legitimate correspondents but financially deterring unwanted correspondents.

Internet telephony. While the channels idea is not easily extended to the traditional telephone network due to the fixed length of phone numbers, it should be usable with Internet telephony [12], since addresses can be arbitrarily long. The channels idea should even work when accessing Internet phone service from a standard telephone set, as long as the call is placed via a server running a PCA that could translate an input phone number or nickname into a channelized Internet phone address. Administration could be via a Web interface or, perhaps, automatic speech recognition. Channelized telephony would allow users to control not only who can call them, but when they can be called; for example, a user could cut off commercial calls during the dinner hour.

Other Approaches

The idea of augmenting the user name portion of an email address with information to aid in routing is not new. The Andrew mail system [2] uses addresses in the form `user+info@host`, whereby `info` is an arbitrary alphanumeric field. Each user may write code in the FLAMES language to process messages based in part on the contents of the `info` field. While the Andrew system could be used to implement the channels approach, it has not yet been used that way. Instead, it has been up to the good will of correspondents not to purposely miscategorize messages, by, say, sending junk mail advertisements to `user+urgent@host`. Such a system, with well-known or easily guessable channels, cannot stand up to the likely onslaught of unwanted email in the commercial world.

Kill files. Another way to avoid email is to automatically discard all messages from a particular user, site, or domain. However, this approach unfairly denies access to legitimate users at the site or domain and is easily evaded through forgery or by having multiple addresses. Channels makes it possible to grant access to any set of individuals, denying access to others, while forgery does not help an adversary evade the channel mechanism.

Email agents and filtering. Email filtering agents [2, 5] can be used to discard messages that fail to satisfy user-defined criteria. However, it is extremely difficult to define syntactic rules that reliably distinguish advertisements and surveys from legitimate messages. Consider the following message, excerpted from one I received recently after purchasing software from the company that makes software package Y:

```
From:frobboz@somewhere.edu (Chuck
Frobboz)
To: hall@research.att.com
Subject:  Difficulty  using  <sw
package X>
Dear Robert,
I have difficulty using <sw pack-
age X> with JR WordProcessor.
[...exposition of some problem...]
Isn't this frustrating? Maybe you
```

```
would like to check out <sw pack-
age Y>. It is really cool. Here is
the URL: [...]

Regards,

Chuck
```

I read several mailing lists regularly where people describe legitimate problems using software packages. This message, really an advertisement, is so similar in form and content to them it would be extremely difficult to write an email filter that reliably discards it but lets through legitimate messages. On the other hand, if this email arrived on a channel allocated to commercial firms, it would have been easy to spot; in fact, a PCA could even demand e-money in advance for a slice of the user's attention.

Cryptographic authentication. A user of cryptography can enforce access control by requiring that all messages be digitally signed by an authorized correspondent; the filter would discard any other messages. If available, this access control method would be an alternative to private channels when messages come from known correspondents, providing good protection against unauthorized messages. However, even though software packages are available to do the cryptographic operations [4, 10], reliably obtaining a correspondent's public key is problematic [11]. Even if this key-certification problem were solved, email software using this access-control method could not deal with messages from unknown correspondents, such as those received from mailing lists. Even messages digitally signed with certified keys are not guaranteed to be not junk. One can accumulate a (large) list of correspondents who send junk, but adversaries can evade this mechanism by registering several addresses and keys or by having different employees send different messages. Channels, on the other hand, allow one to absolutely shut off the flow of messages from an adversary by closing all channels known to it. To gain unauthorized access, users have to invest effort, risk, or money in eavesdropping or social engineering, while new access can be cut off easily once again after just one message.

Legislation. A government might consider extending existing laws governing junk physical mail and telemarketing calls to cover email. However, the global Internet is not governed by a single jurisdiction. Also, legislation would presumably be effective against only law-abiding junk mailers, not harassers and other undesirables.

Conclusions

If people don't know your address, they can't send you email. The channels approach exploits this idea, providing a simple yet effective way to avoid unwanted email. The PCA can essentially automate all the operations necessary to manage the complexities introduced by channels, so routine daily use is transparent to email users. Channels complement cryptographic authentication, because they give control over messages received from unknown correspondents, such as advertisers, survey takers, harassers, and mailing list contributors. In a time of increasing commercialism and decreasing individual privacy, the channels approach shows promise and should be pursued. C

References

1. Blum, L., Blum, M., and Shub, M. A simple unpredictable pseudo-random number generator. *SIAM J. Comput. 15*, 2 (May 1986), 364–383.
2. Borenstein, N., and Thyberg, C. Power, ease of use, and cooperative work in a practical multimedia message system. *Int. J. Man-Machine Studies 34*, 2 (Feb. 1991), 229–259.
3. Costales, B., Allman, E., and Rickert, N. *Sendmail*. O'Reilly & Assoc., Sebastopol, Calif., 1993.
4. Garfinkel, S. *PGP: Pretty Good Privacy*. O'Reilly and Assoc., Sebastopol, Calif., 1995.
5. Greif, I. Desktop agents in group-enabled products. *Commun. ACM 37*, 7 (July 1994), 100–105.
6. Hall, R. Channels: Avoiding unwanted electronic mail. In *Proceedings of the 1996 DIMACS Symposium on Network Threats*. American Mathematical Society, 1997; see ftp://.research.att.com/dist/hall/papers/agents/channels-long.ps.
7. Myers, J., and Rose, M. Post office protocol, v.3. Network Working Group Request for Comments 1725 (RFC 1725, Nov. 1994); see also andrew2.andrew.cmu.edu/rfc/rfc1725
8. Netscape Communications Corp. Netscape Navigator, v2.0 (or later); see home.netscape.com.
9. Postel, J. Simple mail transfer protocol. Network Working Group Request for Comments 1725 (Internet RFC 821, 1982); see www.cis.ohio-state.edu/htbin/rfc/rfc821.html.
10. RSA Data Security. S/MIME Central; see www.rsa.com/rsa/S-MIME.
11. Schneier, B. *Applied Cryptography, 2nd Ed.* John Wiley & Sons, New York, 1996.
12. Vocaltec. Welcome to Vocaltec: The Internet Phone Company; see www.vocaltec.com.

Robert J. Hall (hall@research.att.com) is a principal member of the technical staff in AT&T Labs Research in Florham Park, N.J.