The Internet Protocol Journal

December 2005 Volume 8, Number 4

*A Quarterly Technical Publication for Internet and Intranet Professionals*

## In This Issue



You can download IPJ back issues and find subscription information at: **www.cisco.com/ipj**

## From The Editor

Perhaps the greatest challenge facing the Internet is the ever-increasing amount of unwanted e-mail, commonly known as *spam*. It is tempting to compare electronic mail to its paper counterpart, but there are some important differences. First, "junk-mail" is relatively self-limiting in scope because it costs real money to print and distribute even the most modest flyer. Second, advertisers in the real world are interested in *targeting* their audience. It makes little sense for a supermarket in Boston to advertise weekly specials on produce to consumers in Tokyo. Bulk mail—when delivered by the local postal service—is also quite carefully regulated. It is somewhat rare that you cannot locate the sender of paper-based advertising. None of these observations can be applied to spam. Sending spam is more or less "free," spammers often target "the entire world," and spammers can easily hide behind fake or transient addresses.

To date, spam has been tackled largely by applying sophisticated filtering techniques for incoming e-mail, but this does nothing to decrease the amount of actual spam sent. Anti-spam legislation has been passed in some countries, but it remains difficult—if not impossible—to pursue spammers through legal means, especially in an international context. It is therefore natural to look at technological solutions to the spam problem. If we can secure our network and authenticate its users, would it not be possible to allow only "authorized and verified" senders to send e-mail? Dave Crocker examines this problem in our first article.

Of course, no simple technical solution for spam exists, and not surprisingly there are divergent views on how the problem should be tackled. Our second article, by John Klensin, looks at spam from a different perspective and suggests some possible avenues towards a solution.

Our final article looks at routing protocol testing. Russ White examines testing mechanisms and discusses guidelines for realistic testing.

Many of you have already responded to the *IPJ Reader Survey*. There is still time to participate. If you received an e-mail invitation to take the survey, simply follow the link in the message. You can also take the survey by following the survey link on the IPJ home page: **http://www.cisco.com/ipj**. If you prefer to just drop us a line with your comments and suggestions you can do so by sending e-mail to: **ipj@cisco.com**.

*—Ole J. Jacobsen, Editor and Publisher*
**ole@cisco.com**

# Challenges in Anti-Spam Efforts

*by Dave Crocker, Brandenburg InternetWorking*

It is said that the Internet teaches us one lesson. That lesson is "scaling." The Internet comprises perhaps one billion users, millions of machines and many tens or hundreds of thousands of independent service operators. It operates in, and between, virtually every country on the planet. It is used for personal, organizational and governmental services. Therefore, it must be compatible with many different cultures, many different styles of communication and many different methods of administration. The Internet has no central point of control and operates according to no set schedule. Hence, changes must be gradual and voluntary—when we agree on what those changes should be.

In the early 1990s, the Internet grew from a small research community into a global mass market. Imagine a small town changing into a large, undisciplined city. In a large city, most people are strangers, and the strangers have a diverse range of values and behaviors. Hence, people must use much more caution with each other. In other words, the problems are not with the original way the town operated, but with changing requirements. So, spam is merely an unfortunate—but frankly predictable—example of the Internet's success, not its failure.

This article explores the system-level complexities of the spam problem, as the intersection of social diversity, complexity of e-mail technology and operations, and specific lines of attack that seek to control spam. On the question of control methodologies, most prior work has been on analytic tools that are used by sites receiving spam, to evaluate the mail content, associated addresses or traffic flow. Recent efforts focus on assignment and assessment of an accountable identity that is responsible for individual messages or for the transit of aggregate message traffic.

## The Nature of Spam

People agree that spam is a serious problem, but they have difficulty agreeing on its definition. *Unsolicited Bulk E-mail* (UBE) is probably the most useful.[1] A spammer sends a large number of messages to many different recipients who have not requested the content. (Interestingly most spammers do not care whether a particular addressee receives the message; they merely seek to get a sufficient percent of their postings delivered to some of the addressees.)

Spam can conform to Internet technical standards and can contain no technical differences from legitimate—desired—messages. Hence, spam that violates standards or has other peculiarities might be common today, but detection efforts that are based on these anomalies offer no long-term benefits. Spammers are highly adaptable and use the easiest method that works. However what spam *always* violates are our *social* conventions. Therefore, any long-term, proactive, technical responses to it, such as formulation of standards, must follow, rather than lead our social decisions about it.

Like other social problems, we probably can control spam, even if we cannot eliminate it. This means that we must adjust to having spam as a permanent part of our social landscape, even as we seek to limit it to tolerable levels. Efforts to detect and eliminate spam have been underway for quite a few years. Some techniques have shown useful, localized results, but most only for a short time. In other words, none of the many spam control attempts, over the years, has yet reduced the amount of global spam! So we must be cautious about our expectations for any new anti-spam proposal. It also is likely that controlling spam requires an array of complementary techniques and continued efforts to adapt them, as spammers continue to adapt their own methods. This means that we need to assess any new proposal in terms of its likely *incremental* benefit, rather than as a candidate to be the *Final Ultimate Solution to Solve Spam* (FUSSP).

Changing a global infrastructure takes a long time and is very expensive. Some proposals require complex technology, while others require substantial, on-going administrative effort. Worse, some impose onerous requirements on end-users. Therefore we need to ensure that the mechanisms we deploy will have significant, long-term benefit, even after spammers try to adapt to their presence. They also must have reasonable development cost, require limited, on-going administration and be sufficiently easy to use. In evaluating the likely efficacy of a proposal, a useful heuristic is to ask whether it would be desired even if spam were not a problem. If the answer is yes, then it provides general, strategic benefit, so that counteracting spam merely adds urgency to its adoption.

The Internet provides us all with vastly better access to each other. For collaboration, or the formation of specialized communities or for personal interaction, this is wonderful. For intrusions into our privacy and threats to our online security, this is problematic. Unfortunately, the benefits and the detriments are tightly coupled. Our efforts to control e-mail's problems need to be made cautiously, lest we also reduce its benefits. Worse, our efforts need to limit the damage that might be done to innovative benefits that we have not yet envisioned.

The sender of spam incurs almost no incremental cost for a single message. It is easy to think that we should simply make e-mail be the same as sending letters or making phone calls, by directly charging the sender for every message. This cost provides a barrier against abusive, bulk use. In reality e-mail is a different kind of service, with an extensive history, and it is subject to different choices. Telephones and postal service have highly centralized, formal operational authorities, and the fees charged for their use are based on offsets to direct, real expenses. By contrast, e-mail is a highly decentralized service, with correspondents' private systems contacting each other directly, rather than having to be mediated by state-regulated utilities. If additional fees are charged, they also need to be based on the costs of real services; an arbitrary "tax" will simply create its own problems. For example, who gets the money, and why?

To retain its flexibility and its ability to support new human communication uses, we must retain the current, open model of spontaneous e-mail exchanges. Therefore, over time, it is likely that Internet mail will evolve into two logical subsets. One comprises trusted, accountable participants and the other includes everyone else. Trusted participants may be subject to less stringent checks and filtering. Perhaps more importantly when there is a problem, it is likely that mail from a trusted identity will still be delivered, while the origination agent is consulted, rather than rejecting the mail automatically.

### E-mail Architecture

Internet mail is based on a simple model. It distinguishes the world of users from the world of transmission. Anyone may send a message to anyone else. The basic service does not have a central authority and does not require authentication by the Originator, the Recipient or the operators. (It is worth noting that the telephone and postal services usually do not authenticate those sending letters or making calls.)

As shown in Figure 1, this model has grown to distinguish:

- *Mail User Agents* (MUA), which represent end-users
- The *Mail Transfer Service* (MTS) comprising a sequence of one or more *Mail Transfer Agents* (MTA), using the *Simple Message Transfer Protocol* (SMTP)[2,3]
- Posting new mail via a *Message Submission Agent* (MSA)[7]
- A *Notification Handler* or *Bounce Handler,* is an MUA that processes returned transmission reports such as a notice about failure. The Handler's address is specified by the MSA, during message posting.[11]
- Delivering mail via a *Message Delivery Agent* (MDA), possibly with user-specific delivery behaviors[8, 9]

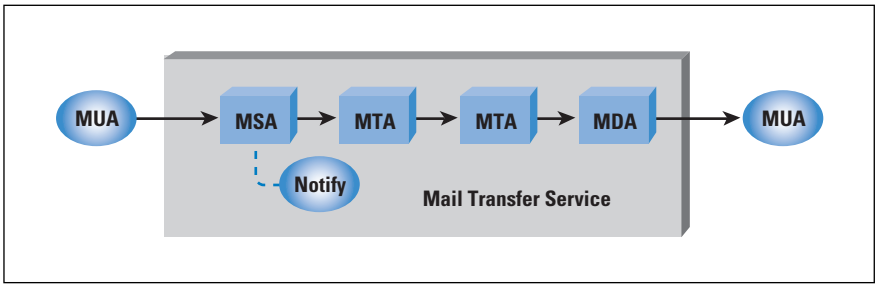



*Figure 1: Internet Mail Architecture*

The purpose of e-mail is to exchange messages among MUAs. For users, their e-mail client—the MUA—is all they directly experience. For most network administrators, the MTS software is their scope of concern.

The core e-mail message object also has a simple framework. Its *content* comprises:

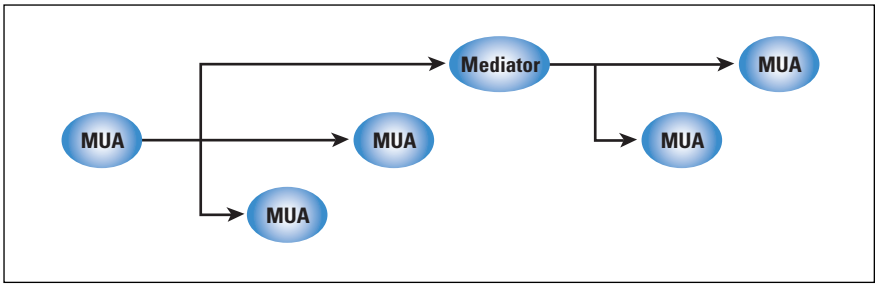
- Structured, textual meta-information, called the *header,* including *fields* for addressing, posting date, unique message identifier and a free-form description of the content[4,5]

- Lines of free-form ASCII text, called the *body*, which has evolved to support a potentially complex, structured set of multi-media, multi-character set attachments[12]

Figure 2 demonstrates a simple user-to-user example, with a message sent to three addressees, one of which is a special MUA that re-mails it to two additional recipients. The purpose of the Figure is to emphasize the user-to-user nature of e-mail and to provide a basis for considering the combinatorial explosion that marks the aggregate interactions of Internet mail components even in very simple uses. It further introduces another architectural construct:

- A *Mediator* is an MUA that re-posts messages, such as for a mailing list.[10] It preserves much or all of the original message, including author address, but can make substantial changes or additions to the content, which an MTA cannot. Therefore, a Mediator's role is user-level content responsibility, rather than MTS-level transit responsibility.

*Figure 2: Simple Multi-Recipient Scenario*

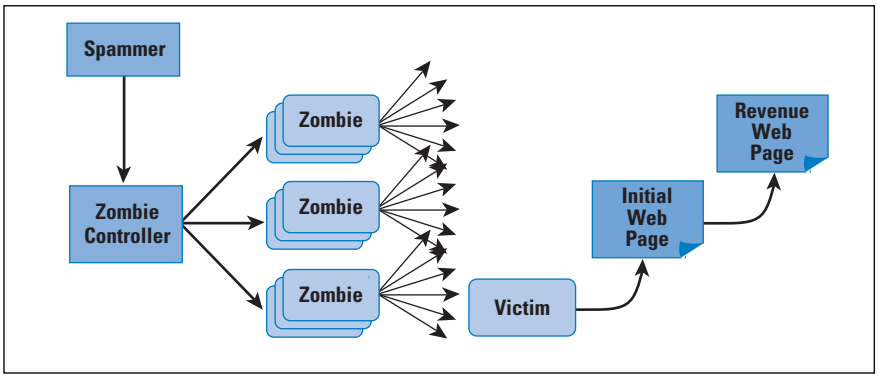



### Spamming Architecture

Some spammers are legitimate businesses, engaged in overly aggressive marketing efforts, because there are no formal limits on their actions. In spite of the challenges created by needing to work at an international level, there is a reasonable expectation that legal strictures, both laws and contracts, will constrain in these businesses to a tolerable level. In contrast, *rogue* spammers actively seek to avoid accountability, to subvert barriers to their traffic, and to acquire unwitting and unwilling participation of machines owned by others. Independent of the legal details, the best social model to use for analyzing this latter group is crime. Often the activities do not violate particular laws, but what is most important is that the style of a spammer's conduct is the same as that of a criminal.

Unfortunately, the technical and operational world of spamming has also developed in scale and sophistication. Spamming used to entail one sender and one sending machine. Its performance was limited by the capacity of that machine and the bandwidth of its Internet connection. Today, rogue spammers control vast armies of compromised systems, called *zombies*, as shown in Figure 3. Zombies are owned by legitimate users who are unaware that their system has been compromised and is being used for spamming.

*Figure 3: Rogue Spammer Control Network*




The community of rogue spammers is remarkably well organized; it has become an extensive, underground economy. Some participants specialize in developing methods for breaking through filters. Others take over machines and turn them into zombies. Others sell the use of a zombie collection for periods of spamming. The estimated number of zombie systems is in the many tens of millions. After spam delivery, recipients often "click" to a transaction Web page. Web hosting is provided at multiple levels, in order to obscure the server side of the process, further reducing accountability.

Typically, spammers have the classic goal of selling products. However, they also can have political or religious motivations or even blatantly criminal intent, such as extortion. The ability to send very large number of messages to a specific destination gives spammers a tool that can be used to threaten an organization with a denial of service attack on their network.

### Practical Efforts at Spam Control

It is tempting to believe that spam is an easy problem to solve, but history teaches us to be cautious. A web page located at **`http://craphound.com/spamsolutions.txt`** takes an irreverent approach in challenging simplistic proposals, by providing a checklist for the common weaknesses. In spite of its apparent whimsy, the checklist is surprisingly useful for screening proposals quickly.

The most common mechanism for spam control is a localized mechanism, the "filter"[14], named for its conditionally permitting mail to flow through it. Filters typically are used within the recipient's network (or Administrative Management Domain, as described later in this article.) However they may be placed anywhere along the path, notably including the MSA. Filters at the reception side cannot reduce Internet spam traffic. At the outbound side, they can. Filters have choices in the way they treat suspect messages. They can:

- Add a special annotation to the message
- Divert it into special storage
- Reject it back to its Handling Notification (RFC 2821 **`MailFrom`**) address or to the Client SMTP during the transfer session
- Simply delete it
- Accept it slowly, with "traffic shaping," to control the rate of SMTP transmission

The difficult question is: What are the criteria that a filter should use? The difficult answer is: Many. This need to support a wide, and changing, variety of decision criteria has caused filtering engines to evolve into extensible platforms for spam detection and handling modules. As the mixture and complexity of filtering algorithms become more sophisticated, the overhead they entail has grown substantially larger.

It is convenient to divide techniques into three, basic classes of criteria, although each is complex:

- *Content analysis,* such as Bayesian statistics tracking of vocabulary and content hashing, to detect bulk duplication
- *Responsible Agent assessment,* either for permission (whitelist) or rejection (blacklist)
- *Traffic analysis,* such as rates at which messages come from the same author address or IP Host Address

Content analysis is always a matter of partial success (and partial failure.) It is usually statistical and depends upon a database of training messages, to establish vocabulary norms. Spammers are constantly developing techniques for bypassing the current analysis technologies. Further, different recipients on the same e-mail service can have wildly different statistical patterns of acceptable content. This makes fine-grained filtering by their service provider problematic.

It is clear that these tools for evaluating individual messages, or aggregate traffic flow, can have significant transient utility. However they cannot be effective, long-term tools, even with continuing enhancement. Notably they have little or no effect at reducing spam at its source. These post-hoc analysis tools have two inherent deficiencies, both of which are coupled to their using heuristics, rather than reliable, accurate and objective rules. The first is one of "false positives" in which legitimate mail is incorrectly labeled as spam. As an example, this could mean that an essential business transaction is not delivered, instead being classed as junk mail. Perhaps the most insidious example of this problem occurs when spammers send mail that purports to be from a well-known, legitimate business. This is called *phishing* and results in making *all* mail with the address suspect, so that legitimate postings of essential mail are not delivered.

The second problem with using heuristics is in the nature of an "arms race" between spammers and anti-spammers who must each constantly adapt techniques, consume more resources, yet never win. It does not help that those fighting spam have been losing the war, since spammers have tended to be more aggressive, more innovative and better organized…

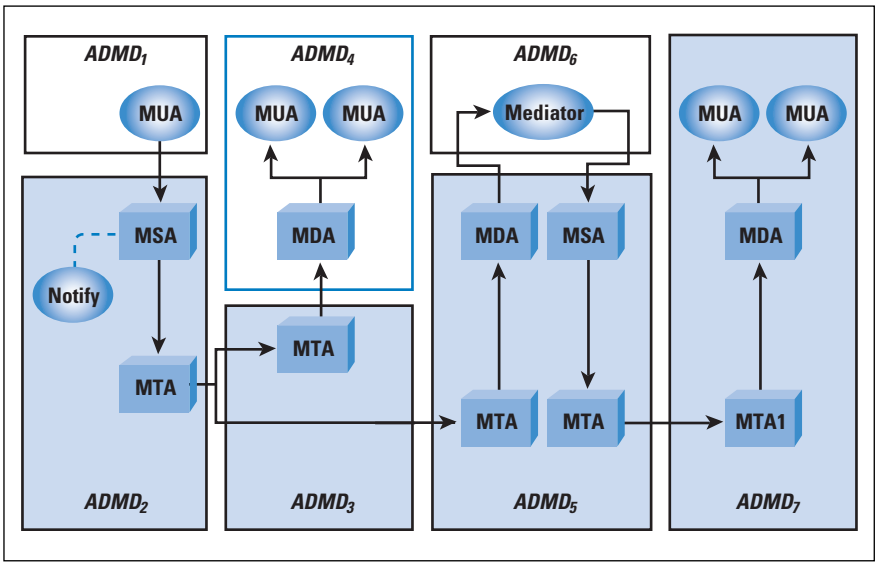
A different line of effort is based on the social assessment that the sender of an e-mail should be held accountable for it. The goal is to identify such an agent and then evaluate the agent's acceptability. This approach requires three enhancements to Internet mail:

- A clear sense of the boundaries between independent operational authorities
- A means of verifying an accountable identity that is associated with the message
- A means of formulating and sharing assessment information about accountable identities

Although e-mail operators often refer to *boundary* MTAs that face the open Internet, there is no accepted term for a region of e-mail components under unified authority. This article suggests a term derived from the OSI X.400 e-mail effort: *Administrative Management Domain* (ADMD) to mark these trust boundaries. They distinguish a collection of operational components subject to the same administrative policies, as discussed in [13].

An example of ADMDs is shown in Figure 4, and is derived from the scenario shown in Figure 2.

*Figure 4: Independent Administrative Management Domains (ADMD)*




The implied complexity of responsibilities and interactions is striking, even for this relatively modest case. For simplicity, think of the ADMDs labeled at the top of the Figure as representing users or value-added services, whereas the ADMDs labeled at the bottom could be a variety of classic Internet service (access) providers. The "boundary" agents are the ones with lines connecting over to another ADMD.

The increased diversity among Internet participants and ADMDs results in abuses such as spam. Proactive efforts to deal with these abuses require that we make changes in the nature of the trust between ADMDs and the way that that trust is enforced.

### Accountability

Agent assessment seeks to hold an entity (agent) accountable for problematic e-mail. Who is a responsible agent for the content or for injecting the message into the MTS, and are they assessed as trusted or problematic?

There are two broad classes of accountable entities:

- *Content agents* comprise authors (RFC 2822 **`From`**) and those who are responsible for posting individual messages, as specified in the RFC 2822 **`Sender`** field. If the content agent is validated for a message, then the content probably reflects their intent. That is, it is unlikely that some other entity changed the content. Because the Notification Handler address (RFC 2821 **`MailFrom`**) appears in the SMTP protocol but is associated with the posting agent, it is often considered useful for analysis. Unfortunately the address often has no obvious relationship to the From field author or the Sender field posting agent, so its use for filtering can be problematic. However spammers often specify false Handling Notices addresses, in order to direct the mass of failed deliveries elsewhere. Consequently, it can be useful to validate the **`MailFrom`** address.
- *Operations agents* provide MTA or basic Internet access services. They are often held accountable for the impact of the bulk traffic their systems generate. Although they do not create the content, it is possible for them to enforce strict rules on their customers and to detect patterns of violations among them. Recommended practices for operators are beginning to obtain some consensus, such as with [15]. More are needed.

Assessment of agents can be proactive or reactive:

- *Accreditation* is the proactive registration by a sender, who aligns with a registry that extracts quality assurance commitments; any trust of the sender is therefore inherited from trust of the accreditation agency.
- *Reputation* refers to reactive evaluation of a sender's prior postings; for these, independent third parties evaluate the sender's history.

The functions that are combined, to establish useful accountability, comprise:

*Identification:* An identity label provides a unique reference to an entity.

*Authentication:* Validates the use of the identity label.

*Authorization:* Determines that the user associated with the identity is authorized to perform a particular function.

*Assessment:* Obtains an analysis of the trustworthiness or "quality" of the agency that is providing the authorization, or of the validated entity itself.

Unfortunately, many identities are involved in e-mail creation or transmission, as shown in Table 1.

**Table 1: Roles for Internet Mail Identities**

| Type | Provided by | Identity of |
|---|---|---|
| MTA IP Host Address | Network-level service | SMTP client |
| EHLO Domain Name | RFC 2821 SMTP command | SMTP client |
| MTA Provider's IP Network Address | Network-level service | Site of SMTP client |
| Mail-From Mail Address | RFC 2821 SMTP command | Handling notices |
| From Mail Address | RFC 2822 header field | Author |
| Sender Mail Address | RFC 2822 header field | Posting agent |
| Received Domain Name | RFC 2822 header field | Relaying MTA site |

Relative to an SMTP Server that is being asked to accept a message, the SMTP Client is an agent of the operator of the previous hop. Since the e-mail operator might be different from the operator of the IP access network that is hosting the e-mail service, it might entail a different identity. This highlights an interesting aspect of Table 1: Most of the identities associated with e-mail handling can be called "the sender." Consequently, that term has become nearly meaningless, in anti-spam discussions.

Because identity listings are made explicitly in a database, they are capable of producing almost no false positives, although there might be many identities not listed and a listing might be inaccurate. Still, there are significant challenges with the use of identity-based filtering:

- Which identity should be used and how does it relate to spamming behaviors? Note that Table 1 listed quite a few choices. In addition an author can create bad content, but the identity listed in the RFC 2822 **`From`** field of that content might not be the actual author, even if that field is validated. The message might have originated on a compromised machine and used the identity associated with it, unbeknown to the owner of the machine. Also the operator of the mail-sending network might have nothing to do with creating content, but it might be reasonable to hold the operator accountable for aggregate traffic problems.
- How is the identity validated (authenticated)? What entity is doing the validation? How does it relate to the identity being validated? And why is it trusted? Can the validation mechanism, itself, be tricked?
- How is an identity determined to be a spammer or non-spammer? What entity is vouching for the quality of that identity and why is the vouching entity trusted?

### Authentication Standards

Accountability requires having an accurate, reliable identity of the agent that is to be accountable. Authenticating an identity is, therefore, a prerequisite for assessment efforts. However it does not, by itself, ensure a positive assessment. Spammers can register and authenticate their identities, too.

Early anti-spam identity schemes use the IP Address of the client SMTP MTA that is sending directly to the server running the filter. The Address is provided by the underlying network service, and therefore has been trusted. However, spammers are becoming proficient at stealing IP Address space, such as by advertising routes that use allocated-but-unused blocks of IP Addresses! Also an IP Address changes as the host changes its attachment to the Internet, and it is affiliated with operators, not authors. This makes the IP Address obscure and unreliable, when attempting to assess e-mail.

A more recent focus is on the use of Domain Names, for references that are more stable and align better with the authority boundaries of Administrative Management Domains. Broadly there are two lines of effort at using Domain Names for validating messages being relayed. One associates the identity with the systems that handle the message along its path. These "path registration" schemes include Sender Policy Framework, Sender-ID, and Certified Server Validation. The other schemes tie a Domain Name identity to the message object. These include DomainKeys Identified Mail, and Bounce-Address Tag Validation.

The *Sender Policy Framework* (SPF)[16] has evolved over time, attempting to encompass multiple identities. It primarily uses the Domain Name in the RFC 2821 **`MailFrom`** command. It queries the *Domain Name System* (DNS) with that name and determines whether the IP address of the previous-hop MTA is registered under that name. Since any SMTP server along the transit path may choose to perform this query, SPF requires that the Domain Name contain a registration for every MTA along every delivery path for a message. (A common simplification for this model is to use it only between boundary MTAs, but this considerable constraint is not specified in SPF. Rather, its use is usually characterized as being more general.) Although the software overhead for SPF is quite small, the administrative overhead can become substantial, as the number of paths increase and as paths change. In addition, some sender SPF DNS configurations can trigger a very large number of queries per addressee. Lastly, the role of the RFC 2821 **`MailFrom`** command is to specify the Notification Handler address. This address might be entirely different from other origination information, making registration of all of the MTAs in the path problematic. SPF therefore has significant administrative problems with redirected traffic, such as when going through a third-party forwarding service.

*Sender-ID* (SID)[17] uses a model similar to SPF, but it is based on the posting address Domain Name in the RFC 2822 **`Sender`** field (or RFC 2822 **`From`** field, if no **`Sender`** field is present.) Both SID and SPF sought IETF standardization in 2004 but the working group effort failed, due to lack of rough consensus convergence among participants and due to concerns over intellectual property claims.

*Certified Server Validation* (CSV)[18] covers only the current client/server SMTP hop. The client specifies an operator's Domain Name in the RFC 2821 **`EHLO`** command. The server uses this name to query the DNS. It then validates the IP Address of the SMTP client and determines whether the Domain Name administrator has authorized the client to send mail. CSV also specifies a standard mechanism for querying an assessment service about the client's Domain Name.

*DomainKeys Identified Mail* (DKIM)[19] specifies an accountable Domain Name that applies to a message during transit. It uses public key cryptography to digitally sign the message and provides guidance when the signing Domain Name differs from the Domain Name in the RFC 2822 **`From`** field.

DKIM Domain Name validation represents a significantly different goal from that of the strong authentication methods, such as [20, 21] which focus on long-term protection of message content. Also DKIM places its parametric information in a special RFC 2822 header field, rather than in the message body, so that it does not have any impact on recipient user agents that do not support DKIM. Although public key cryptography has relatively high computational cost, e-mail processing is usually i/o-bound, so that the real-world use of DKIM appears to have little impact on the aggregate message-handling capacity of a server.

*Bounce Address Tag Validation* (BATV)[22] attacks the problem of misdirected handling notices, such as bounces. It permits the creator of an RFC 2821 **`MailFrom`** bounce address to digitally sign it. When the bounce agent of that creator receives a message purporting to be a bounce, the agent can validate the address. Standardization of its format is needed so that e-mail intermediaries—such as some mailing list software—can determine the "core" of the mailbox portion. Since the creator of the signature semantics is the only consumer of the signature semantics, any signature algorithm can be used, including one based on symmetric keys. For convenience—and an existence proof—the BATV specification provides an example algorithm already in use.

### Collaboration Support

Fighting spam must be a collaborative effort, which will benefit from using tools and standards that aid in exchanging information and performing coordination. To this end, standard methods of reporting spamming events, of characterizing particular spam, and of sending spam control data can be helpful. Some work in that direction is already underway.[23] Fighting spam requires global operations collaboration; this will be aided by services to facilitate interactions between network administrators speaking different languages. It is also likely that there should be standards for the syntax and semantics of whitelists and blacklists.

## Acknowledgement

The author wishes to express particular appreciation for the unusual amount of dialogue that took place with the reviewers of this article. It produced a substantially clearer and more concise article. It also highlighted the extraordinary diversity of views on the topic, in case one had had any doubt. In fact, the article by John Klensin which follows this one is a direct result of the dialog.

## References

[1] Hoffman, P. and D. Crocker, "Unsolicited Bulk Email: Mechanisms for Control," Internet Mail Consortium, UBE-SOL IMCR-008,
**http://www.imc.org/ube-sol.html**, revised May 4, 1998.

[2] Postel, J. B., "Simple Mail Transfer Protocol," STD 10, RFC 821, August 1982.

[3] Klensin, J., "Simple Mail Transfer Protocol," RFC 2821, April 2001.

[4] Crocker, D.H., "Standard for the format of ARPA Internet text messages," STD 11, RFC 822, August 1982.

[5] Resnick, P., "Internet Message Format," RFC 2822, April 2001.

[6] Crocker, D., "Internet Mail Architecture," Internet Draft, **draft-crocker-email-arch**, April 2005.

[7] Gellens, R. and J. C. Klensin, "Message Submission," RFC 2476, December 1998.

[8] Myers, J. G. and M. T. Rose, "Post Office Protocol – Version 3," STD 53, RFC 1939, May 1996.

[9] Crispin, M., "Internet Message Access Protocol – Version 4rev1," RFC 3501, March 2003.

[10] Chandhok, R. and G. Wenger, "List-Id: A Structured Field and Namespace for the Identification of Mailing Lists," RFC 2919, March 2001.

[11] Moore, K., "Simple Mail Transfer Protocol (SMTP) Service Extension for Delivery Status Notifications (DSNs)," RFC 3461, January 2003.

[12] Freed, N. and N.S. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies," RFC 2045, November 1996.

[13] Clark, D., Wroclawski, J., Sollins, K., and R. Braden, "Tussle in Cyberspace: Defining Tomorrow's Internet," ACM SIGCOMM, 2002.

[14] Showalter, T., "Sieve: A Mail Filtering Language," RFC 3028, January 2001.

[15] Hutzler, C., Crocker, D., Resnick, P., Sanderson, R., and E. Allman, "Email Submission: Access and Accountability," Internet-Draft, **draft-hutzlerspamops-05**, October 2005.

[16] Wong M., Schlitt M., "Sender Policy Framework (SPF) for Authorizing Use of Domains in EMAIL, version 1," Internet Draft, **draft-schlitt-spf-classic-02**, June 2005.

[17] Lyon J., Wong M., "Sender ID: Authenticating Email," Internet Draft, **draft-lyon-senderid-core-01.txt**, May 2005.

[18] Crocker D., Leslie J., Otis D., "Certified Server Validation (CSV)," Internet Draft, **draft-ietf-marid-csv-intro-02**, February 2005. Also see: **http://mipassoc.org/csv**

[19] Allman E., Callas J., Delany M., Libbey M., Fenton J., Thomas M., "DomainKeys Identified Mail (DKIM)," Internet Draft, **draft-allman-dkim-base-00**, July 2005. Also see **http://mipassoc.org/dkim**

[20] Ramsdell B. (ed.), "Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 3.1 Message Specification," RFC 3851, July 2004.

[21] Elkins M., Del Torto D., Levien R., Roessler T., "MIME Security with OpenPGP," RFC 3156, August 2001.

[22] Levine J., Crocker D., Silberman S., Finch T., "Bounce Address Tag Validation (BATV)," Internet Draft, **draft-levine-mass-batv-00**, September 2004. Also see **http://mipassoc.org/batv**

[23] Shafranovich, Y., "An Extensible Format for Email Feedback Reports," Internet Draft, **draft-shafranovich-feedback-report-01.txt**, May 2005.

*Ed.:* This article is a revision of "Adapting Global Email for Controlling Spam," in *Information Processing Society of Japan (IPSJ) Magazine—Special issue on Anti-Spam,* Japanese/English, Volume 46, No. 7, pp. 741–746, July 2005.

DAVE CROCKER is a principal with Brandenburg InternetWorking. He has authored or contributed to most Internet mail standards, and an assortment of e-mail products and businesses, as well as working on facsimile, security, ecommerce and EDI. He received the 2004 *IEEE Internet Award* for his work on e-mail. Dave is a contributor to the development efforts for DKIM, CSV and BATV, motivated by a strong desire to protect more than 30 years of professional investment that is being threatened by spamming. E-mail: **dcrocker@bbiw.net**

# Taking Another Look at the Spam Problem

*by John C. Klensin*

The problem of unsolicited bulk e-mail on the Internet has been widely discussed, and many classes of solutions have been proposed. Dave Crocker's article discusses some of the background for the solutions generally, points to a semi-humorous list of ways in which proposed approaches fail, and compares several approaches based on source authentication. This article takes a somewhat contrarian view. It argues specifically that the traditional models for defining technological solutions and then letting the policy and legal communities work out the details of how to utilize them are seriously wrong in this particular case and that partially-effective methods of fighting spam actually cause more spam.

This article makes two main suggestions. First, attempts to design technological countermeasures to spam without a clear understanding of how far, and in what directions, the setters of social policy are willing to go are futile. The requirement is not just that there be social recognition that a problem exists. In order to design effective technological countermeasures with predictable and acceptable side-effects, we must first understand what measures society is willing to take—what laws it is willing to pass and enforce to make spam a criminal or civilly-punishable act—to set an appropriate context and set of boundary conditions. Without those conditions, design of technological countermeasures is likely to constitute poor engineering practice, not just futility. Second, deployment of spam counter-measures that are not completely effective largely shifts the burdens of spam from one recipient population to another while *increasing* the total amount of spam on the network.

His analysis and mine agree on several critical points. Solutions that discard important characteristics of today's e-mail environment permanently in order to make some short-term gains against spam are not acceptable. Approaches that require drastic and simultaneous changes to the ways in which e-mail works in order to function are not going anywhere. There is a difference between legitimate businesses who have decided, within the limits of existing legislation, to engage in mass, unsolicited, electronic mailings to promote their products and those bulk mailers who prefer to cover their tracks, hide linkages between sending addresses, hosts, and web sites (or create deceptive ones), and who use zombie mailers and other ways to avoid cost and detection. We also agree that spammers, or their tool suppliers, are creative, technically-knowledgeable, and able to react much more quickly than the spam-fighting community (especially the standards-based part of that community) to changes in operating conditions and countermeasures.

I suggest a further guideline to help us think about the problem: however small they might be on a per-message basis, there are costs associated with sending e-mail and costs associated with receiving it and eliminating undesirable content.

If an anti-spam "solution" is developed that permits the spammers to vastly increase the costs to the recipients without a proportionate increase in their own costs, that solution is not tenable. A serious effort to predict the impact of a proposed solution to spam, including costs to the end user and load on the network as the spammers adapt to it, should be a critical component of such efforts. But, while equivalent analyses of measures, likely responses, and countermeasures are standard with any (other) technique designed to enhance network security, they have been largely absent when new technological approaches to spam are proposed.

This is a different aspect of the so-called "arms race" problem. In a classic arms race, no one can really win, as Dave points out. But, more important, when such races stop, it is only because one party simply stops, is forced out of the game by external pressures, or becomes exhausted economically. As long as there are no economic constraints, every escalation is met with a counter-escalation, which is met with a counter-counter-escalation, and so on. It is this positive feedback cycle that characterizes a true arms race. The battle against spam demonstrates a particularly unfortunate variation on that pattern in which the incremental economic costs of trying to deploy new spam abatement measures appear to be much more severe than the costs to the spammers of the most obvious counter-measure to improved spam abatement procedures, simply sending out more traffic. This is discussed further and in context below.

### Social Problems and Technological Solutions

In the technical and protocol design community, our normal model is to develop technology and then use it to inform the policy, social, and legal parts of the society who then need to sort things out on their side. One of the classic arguments for this approach, which does not seem relevant to the spam situation, is that the potential use or misuse of a technology will not, and should not, constrain its development. For spam, the situation appears to be exactly reversed: we need to understand what is feasible and plausible from social, political, legal, and regulatory standpoints in order to define the engineering solution space. If we do not know what behaviors society is willing to make illegal or subject to effective civil action and whether it is willing to enforce those laws or equivalent positions, we cannot adequately define the engineering solution space. That results, in turn, in a high risk of solving the wrong problem or an irrelevant one. Of course, recent history has shown a variety of irrelevant and costly solutions to spam proposed, and sometimes deployed.

The solution to spam is identical to the solution to most other significant social problems: society must determine that it is a problem, create effective rules prohibiting the problem, and then enforce those rules aggressively and consistently. Technical solutions that make it easier to identify spam and its sources can then be immensely useful, but they are only useful if designed to be effective within the framework set by those rules.

If, by contrast, societies are, in practice, unwilling to take effective social or legal action against spam and those who benefit from it, then this article suggests that anti-spam measures will tend to make the overall situation worse.

The question of spam beneficiaries provides a particularly good illustration of this point. So far, most legal systems in the world have taken the position that the act of spamming is the offense (if there is any offense at all). Operating a domain or web site to which the spam recipient is directed to buy a product or obtain another benefit is rarely considered a problem by either law enforcement or by the relevant ISP. While establishing cause and effect—that the spam was authorized or encouraged by the web site owner—can be quite difficult, there has, appropriately, been little examination of tools to detect or identify beneficiaries because doing so seems pointless. On the other hand, on the same theory that it is more useful to try to arrest the drug importer than the street dealer, a different set of laws about beneficiaries and spam-authorizers—those who, in at least some cases, pay the spammers to spam—might dramatically change the landscape.

### Reducing Spam by the Percentages

A new technique or group of techniques that claims to be beneficial can have either positive or negative value with regard to the amount of spam that gets through, either overall or to the mailbox or a particular sample user. A technique can also result in significant increases in the amount of network bandwidth or server resources consumed if it is neutral or better with regard to the end user mailbox. As long as the spammers can increase the number of messages they send out, almost arbitrarily and at low or zero marginal cost, the percentage of spam that is filtered out is ultimately irrelevant. The key measurement is how the amount of spam that gets through to some exemplar user (or a statistical aggregate of them) changes. That change pattern can be net either positive or negative. Suppose a technique is introduced that causes an initial small incremental reduction in the amount of spam delivered. The patterns of the last several years suggest that the spammers will respond by making a large increase in the amount of traffic they send out. Since the costs of doing so are very low, it would arguably be irrational for them to do anything else. If the increased volume is enough larger than the amount of spam the new technique was able to stop, there is a net loss to the Internet overall: the small improvement may represent a percentage decrease in the amount of spam that gets through, but the amount seen by the representative user increases and the percentage claims are largely irrelevant.

Unless whatever methods that are used in an attempt to reduce the amount of spam actually stop it at, or very near, the point of origin, the net effect on users is to shift the amount of spam received from those who have deployed the latest and most effective countermeasures to those who have not yet done so. The total amount of spam-related traffic on the network just continues to rise. And, since most countermeasures have costs—either in processing time or in software licensing fees—the cost burdens on end users also continue to rise.

This would seem to argue for methods that cut off spam traffic close to the source, but attempts to design such methods have been fairly unsuccessful, sometimes because of another policy problem: the spammers argue that some people like receiving unsolicited bulk commercial e-mail so that cutting off bulk traffic near the point of origin prevents legitimate and desired traffic from transiting the network. Source-oriented techniques include not only technical approaches but efforts—by law or social pressure—to hold ISPs and mail providers responsible for all traffic emanating from their networks, thereby encouraging them to refuse to have spammers as customers, to aggressively enforce terms and conditions of service, and so on. The strongest advocates of the "blacklist" variation of those techniques continue to claim that they are very effective although some others in the community are not completely convinced.

### The House-Burglar Analogy

In the absence of a coordinated approach that is oriented toward legal or social enforcement, most anti-spam techniques appear to induce more spam on the network. They do this by making simply sending much more traffic out the most rational behavior for a spammer who is faced with an abatement technique to adopt. They may enable shifting the burden of dealing with that spam from one person to another—in the same way that aggressive locks and alarm systems on one house slightly increases the relative burglary risk to the less-protected neighbor—but, as Dave's article points out, we have no realistic plan for making it too expensive for the spammers to simply increase output.

Deterrents to burglary work moderately well because they increase the costs (in time, sophistication of the required tools, and so on) to the burglar. Equally important, they increase the risks of being caught and punished. In the present spam environment in most countries, we have no effective mechanism to increase costs and, at least statistically, the odds of being effectively punished even if caught are insignificant.

### Shifting Burdens and Creating Preferred Classes of E-mail

The argument Dave presents for authenticated mail is ultimately that it can get expedited handling while non-authenticated mail is put aside for other methods of spam detection. That approach could be immensely effective at expediting receipt of some mail by the recipients who apply the needed checks, at least until the spammers begin authenticating their mail in a way that tricks the trust-establishment techniques. Prioritization of some messages and content will be effective as long as the fraction of such messages remains relatively small relative to the total number of messages received. As the percentage rises, one probably ends up either trusting all mail from a particular source, regardless of the author, or with a situation quite analogous to "whitelists," although one that is much harder to trick than the original. Either is subject to attacks and scaling problems.

There is also the risk of abuse by providers who conclude that mail that cannot be authenticated well enough that their users can prioritize it should simply be rejected and who then define the conditions for adequate authentication in terms of a small circle of cooperating mail providers. Even if the types of authentication outlined in Dave's article are used only as intended, the costs to recipients will rise, perhaps rapidly, over time as percentages of messages bearing authentication information rises and sender authentication and authorization become just one more tool to distinguish probably-desired messages from probably-undesired ones.

### Maybe there is not Enough Spam Yet

One of the depressing consequences of the reasoning discussed previously is that perhaps we have yet to see sufficient spam for governments and regulatory bodies to take the spam problem seriously—seriously enough to deploy effective laws and enforcement mechanisms. If spam-fighting methods shift the burdens of receiving spam away from those who have the resources to protect themselves they may simply place the spam impacts on others who have fewer resources. That pattern may, in turn, also reduce pressure on governments to take effective action and to do so in a way that would make the design constraints for effective technological approaches clear. If a collection of anti-spam methods have the effect of simultaneously increasing the amount of total spam on the network and of decreasing pressures on societies and governments to take effective action, are they really ones we want to deploy?

### Conclusions

This article presents a rather grim view of the future if we continue on our present course. If we fail to examine the actual actions that societies and their governments are willing to take to deal with spam and spammers and to treat those actions and their limitations as design constraints on the technical and engineering approaches, we are likely to continue to see an ever-increasing amount of spam on the network. Spammers will not only adopt technical countermeasures to new techniques but they will also take advantage of their ability to simply increase message volumes (at almost no cost) to counter the effects of those techniques on the percentage of spam that is delivered. It may be time to finally deal with the spam problem as the difficult social issue that it is, rather than permitting societies and governments to continue to believe that a technological "silver bullet" is right around the corner and that no real social or political action, or commitment of law enforcement resources, is needed.

JOHN KLENSIN is an independent consultant based in Cambridge, Massachusetts. He has been involved in the design, development, and deployment of ARPANET and Internet applications, and occasionally lower-layer technologies, since the late 1960s and early 1970s. He has also been intermittently involved with Internet administrative and policy issues since the early 1980s. His current work primarily focuses on internationalization of the Internet on both technical and policy dimensions. E-mail: **klensin@jck.com**

# Caveats in Testing Routing Protocol Convergence

*by Russ White, Cisco Systems*

In general, the main problems we find when testing routing protocols lie in generating accurate (or rather, realistic) data, as well as understanding the limitations of tests geared towards measuring routing protocol performance. Three areas of specific interest are covered in this article: defining convergence, taking realistic measurements, and creating realistic data.

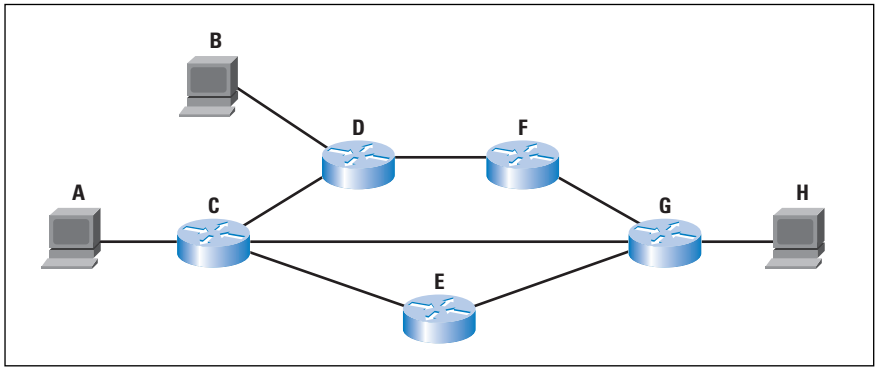
## Defining Convergence

The first problem we face when trying to test routing is to define *convergence*. It seems like a simple question, but it's not, because there are so many different ways to measure convergence:

- How long does it take to begin forwarding traffic once a topology change has occurred?
- How long does it take for every router in the network to adjust to a topology change that has occurred?
- How long does it take for the forwarding information on a specific router to be updated once a topology change has occurred?
- How long does it take for the routing protocol to adjust to a topology change?

Each of these questions is actually completely different, as a short examination of the network in Figure 1, below, shows.

*Figure 1: Test Network*




Assume A is the traffic source for a test, and H is the sink, or the convergence measurement point. To measure the convergence time of this network, you send a stream of traffic from A to H; when the traffic stabilizes, the C to G link is taken down, and the length of the gap in traffic at H is measured. In this environment, we assume the path fails off of the C to G link, and onto the path through E.

This test assumes the traffic between B and H, or between A and B, will not be impacted by the link between C and G failing, but we do not know this will always be the case. In fact, it's possible that D and F will end up forming a *microloop* until they receive all the information needed to converge without the C to G link.

This microloop could last longer than C requires to recompute a path to H, so while the traffic from A to H may be successfully delivered, the network may not be in a fully converged state. The topic of microloop formation and avoidance is beyond the scope of this article.

In this small network, the time it takes for A to continue forwarding traffic to H may not be the same as the time it takes for the entire network to stabilize after the topology change. How long it takes for A to be able to reach H, and how long it takes for all the routers in the network to adjust to the topology change are two different questions. In this case, the concept of convergence is unclear, with several possible meanings; to properly build and understand the results of the test, we need to better understand the question being asked.

You could alter the test so only A, C, E, G, and H are in the network. This would provide a "clean" test of just the failover capabilities of the routing protocol being tested, as it's implemented on the specific routers in the network, across the specific link types connecting the routers, in the simple failover situation. While the limited topology does limit the number of outputs being measured in the test, it also limits the closeness of the tested network to a real network design. The test can provide some very specific data points, but, once the test topology is simplified, it cannot provide a true picture of convergence in a larger, more complex topology.

Another option is to refine the test procedure so the traffic between B and H is tested as well as the traffic between A and H. Measuring traffic flow from every possible connected end point to every other possible connected end point on the network provides a number called *goodput,* which is the relation between the traffic injected into the network versus the traffic the network delivers across all paths.

Although this type of testing does provide more data in a more complex topology, it also has its drawbacks. For instance, if you are trying to compare two different implementations of a single protocol, or compare two different routing protocols, this test not only counts the amount of time required for the routing protocol to converge, it also tests the amount of time required to note the topology change, the time required to install the newly computed routes into the local routing table, and the time required to pass the changes from the routing table to the local forwarding tables. This might—or might not—be a good thing.

Isolating just the routing protocol can provide information about the performance of a specific implementation of the protocol in specific network designs, and under certain conditions. Including platform and media-specific issues—such as the installation of information into a local table—may cloud the picture. For instance, if the routing protocol can converge in milliseconds, but it takes seconds to determine that the link between C and G has failed, any changes in routing protocol convergence time will be lost in the much larger link failure detection time, reducing the value of the test.

In short, numerous tradeoffs are involved in designing a test to measure routing protocol convergence; you need to begin with the right questions, and understand the tradeoffs in the various tests you could, or might, run. There's no "simple" way to run a single test that will give you all the information you need to know to understand all possible implementations of a routing protocol on all possible platforms.

In the same way, it's important to keep these types of limiting factors in mind when reading, or using, test results provided by outside companies. It's fairly easy to look at a specific test for one measure, such as the number of neighbors a specific implementation of the *Border Gateway Protocol* (BGP) can support in specific conditions, and attempt to generalize those test results to much larger and varied real world networks. Quite often, the mapping isn't all that simple.

### Taking Realistic Measurements

Assume you determine you want to test for protocol convergence by checking the routing tables at each router in the network in Figure 1, rather than trying to measure convergence by measuring traffic flow through the network. How would you go about doing this? There are two general types, or classes, of tests, that you could consider:

- *Black Box:* Treat the device as a black box, only using outside signals and controls, and never any output provided from the device itself.
- *White Box:* Use available output provided from the device itself, possibly with tests using signals outside the device, to determine when specific events on the device occur.

Obviously, black box testing is much more difficult, maybe impossible in some conditions, but, at the same time, can provide more "objective" measures of a devices' performance. Examples of black box tests for the *Open Shortest Path First* (OSPF) protocol are outlined in RFC 4061, RFC 4062, and RFC 4063. White box testing typically depends on *debug* and *show* commands to provide timestamped infor-mation about when specific events occur, such as when the routing protocol has received information about the topology change, when the routing protocol has finished computing the best path to each destination, and other events.

For simplicity, the network is reconfigured with a test measurement device, as shown in Figure 2, below.

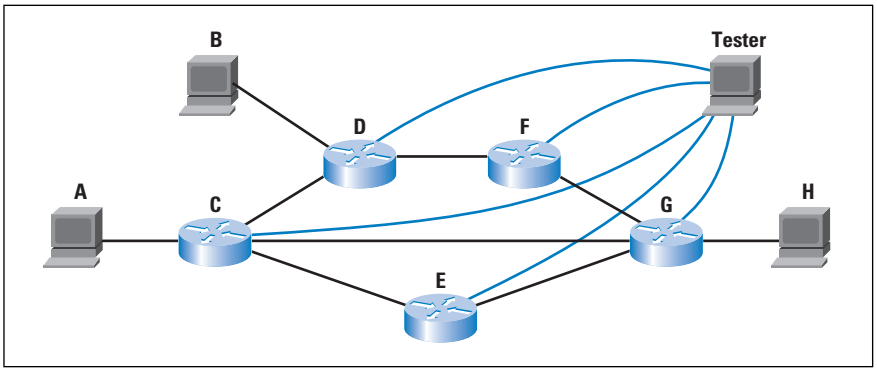



*Figure 2: Reconfigured Test Network*

Some mechanism is used to determine when the routing protocol on each router has computed the correct routes; the network is connected, and allowed to converge. The link between C and G is taken down, and the time between the link failure and the correct routes being computed on C, D, E, F, and G is taken as the total convergence time in the network. This appears to be a straight forward test; what sorts of problems can we run in to here?

There are two possible mechanisms for determining when each device has correctly computed the routes after the C to G link fails:

- Some sort of "continuous output," such as a *debug*, can be configured on each router, and the results collected and analyzed.
- The Tester can poll each device, using *show* commands, or some black box testing technique, to determine when device has recalculated the routes correctly.

Let's examine each of these techniques separately.

#### Gathering Results from Continuous Router Output

The first, and simplest, mechanism is to gather the results from each router through debugging information provided by the protocol implementation which is generally used for troubleshooting and monitoring the routing protocol. There are three primary issues related to using this information you need to be aware of:

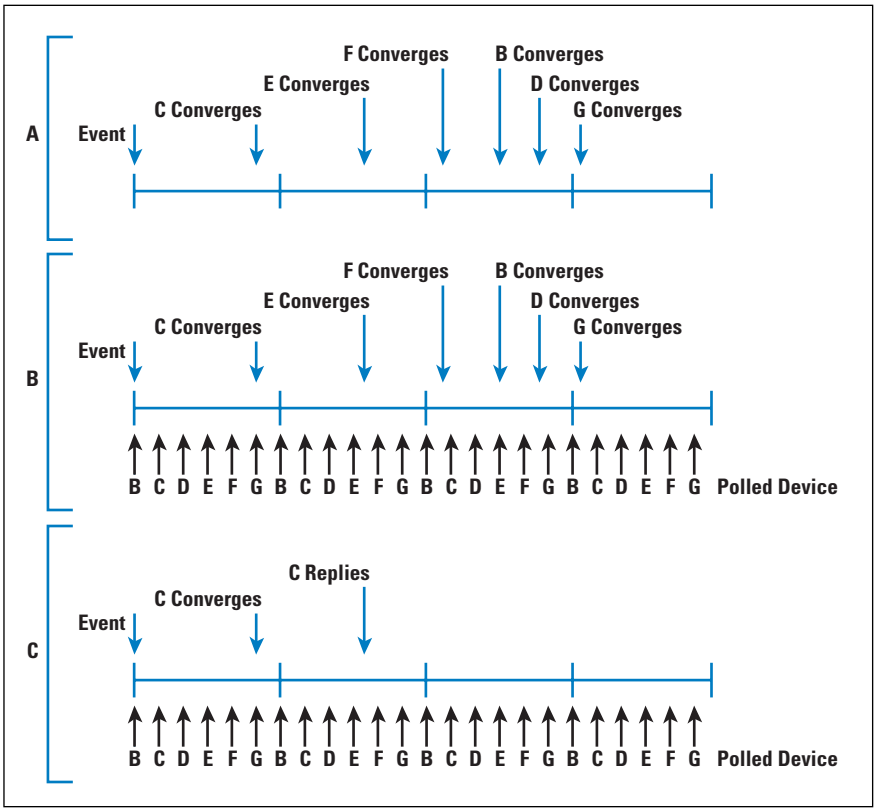
- The continuous stream of information provided by the device being tested can actually impact the test results, primarily because of the processor cycles required to record and display this information. In some situations, the additional cost is negligible, and in others, it's simply not important (for instance, if the test is designed to show the differential between two situations, rather than provide absolute convergence times).
- If the timestamps injected by the devices being tested in the network are relied on, then the time clocks of every device must be synchronized. This synchronization must generally be within about 1/10$^{th}$ or less of the total variation in the test time for the results to be meaningful. In other words, if the timeclocks on all the devices are synchronized within one second of each other, and the results of the test are expressed in milliseconds, the actual test results are going to be lost in variations in the synchronization of the timeclocks.
- If the devices feed their information to the Tester, and the timestamp on the Tester is used to compare the event times within the network, the timestamps can be skewed by the packet processing requirements of the devices, as well as queuing delays in the Tester. Most routers prioritize routing traffic over switched traffic, and switched traffic over management traffic. There could be significant lags between an event occurring, and the router actually building a packet noting the occurrence of that event. Again, this is a matter of time differentials; if the test results are expressed in milliseconds, queuing delays alone can bury the results in noise.

We need to be careful when using *debug* or other continuous output to measure network convergence times in any given test, then. Quite often, we need to compare the granularity of the test results with the measurement technique used, and consider how much noise the measurement technique is actually likely to inject into the testing environment, compared to the test results granularity.

### Polling Devices

Another common technique is to run some sort of process on the Tester which polls each device, either using some black box or white box measurement, to determine when each device finishes recalculating routes after the topology change has occurred. This type of test is also constrained by various factors that might not be obvious when you are designing a test, or examining the results of a test that uses it. Assume events in the network occur as Figure 3 illustrates.

*Figure 3: Poll Testing Scenario*




In Figure 3A, we assume that the Tester is able to poll every device in the network at the same time, once a second. The test shows the network converged at 4 seconds after the event, although the last router to converge, G, does so just after the 3 second mark. There can be a variation of the entire polling interval in the actual results without the test showing any difference in the convergence time of the network, implying that the polling interval must be much faster than the expected (measured) test results for the results to be meaningful. We normally suggest that the polling interval be about 10 times faster than the expected measurement rate, or that the Tester should poll every 1/10$^{th}$ of a second in this test, if the results are to be measured in seconds.

However, in real test environments, a test device cannot actually poll every device in the network at the same time. Instead, the Tester will poll one device periodically, rotating through the polled devices, so the longest time between any specific device being polled is the polling rate. We can call this rotating polling *serialization,* and the time it takes to rotate through all the devices the *serialization delay*. Here, we've spread out the polls across the total one second polling time, to illustrate, in Figure 3B. Three anomalies show up in this illustration:

- The total time for the network to converge is still just over three seconds, while the recorded test time is still in the four second range. This is similar to the problem we noted when we assumed the Tester was polling all the devices in the network at the same time.
- It appears, from our test results, that E and F have converged at about the same moment. In reality, their convergence is separated by almost one second. In some extreme cases, the devices may actually converge in the opposite order from the order they appear to converge.
- If the convergence order of D and G were to be reversed, the network would appear to converge almost a half a second faster, although the actual convergence time would remain constant. This could cause a widely diverging set of test results over multiple runs in what is, actually, a fairly consistent network convergence time.

Adding the serialization delay of polling isn't enough, however, to understand polling in real test environments. We also need to remember that each device which is polled must also answer each one of the polls, thereby introducing another variable amount of delay into the test results. For instance, in Figure 3C, C is polled once before and once after it converges. If we take the time that C answers as its convergence time, then we are also including processing time on C, which is variable, into C's total convergence time. However, if we take the polling time as C's convergence time, it's possible that the poll was received before C converged, and was processed, and answered, after C converged, skewing the results in the opposite direction.

Unfortunately, there are no simple answers to these problems. Instead, when you are designing a test, or examining the results of a test, the mechanism used to determine convergence, the rate at which that mechanism is used, and the reported final results, should be taken together, and considered closely. A test which reports results in milliseconds, but polls a large number of devices from a single test device, should be examined closely for serialization delay errors.

#### Use Real-Life Configuration Parameters and Prefix Attributes

Finally, we need to consider what is probably one of the most widely disregarded concerns in testing routing protocol implementations: building accurate and repeatable data sets to feed into the test. Let's examine a common test, to help in understanding this problem.

A network engineer sets up a router connected to a router testing device using a SONET link. The router tester is then configured to feed one million routes, through BGP, to the router being tested. The test is run, and the amount of time it takes for the router to accept and install all of the routes into its local tables is measured. The router is disconnected (we'll call this first router A), and another router (B) is connected. The same test is performed. In the end, the network engineer proclaims A has a better BGP implementation than B, because A accepted and installed the routes fed to it faster than B.

This sort of test, and these results, should raise a lot of red flags for anyone who's ever tested routers before. Many questions here are not answered:

- Were both routers tuned to optimum parameters for this specific test? Most routers are installed in a number of different situations in various networks, and most will perform better if they are tuned to fit the role they are playing in the network. This is similar to tuning a server for database use, or web server use.

- BGP is very sensitive to the data transmitted from one router to another; BGP implementers are generally aware of this, and use differing models of BGP behavior in different networks to tune their implementations. Specifically, in the case of BGP:
  - What percentage of the prefixes transmitted were of specific prefix lengths? What percentage of the routes transmitted were /24s, /23s, and so on?
  - How many different attribute sets were represented in the routing information transmitted? What number of unique attribute sets were included in the routes? For each attribute set, what percentage of the table did that attribute set represent?

Each of these questions can, and should, be compared to real world measures in the network the router is going to be installed in. There are some instances where protocol implementers have tuned their implementation for use in an Internet *Point of Presence* (POP), for instance, and the implementation doesn't fare as well as a route reflector, or the other way around. For some vendors, this tuning could even be on a platform by platform basis, making the job of characterizing a specific implementation through a simple test, like that described above, very difficult.

### Conclusion

Designing, executing, and evaluating the results of a test attempting to measure network convergence is much more complex than it appears on the surface. In any given test situation, we need to ask:

- What was the test designed to measure? Is it measuring the appropriate outputs, in the correct ways, to actually measure this?

- What is the granularity of the test results and the actual network events, compared with the measurement techniques used in the test? Will normal test results get lost in the noise introduce by the measurement techniques?
- What is the data set used to build the test? Does it accurately reflect the data the routing protocol implementation will be handling in a real network (or more specifically, the real network the router will be installed in).

When designing, or evaluating, test results, there's a strong tendency to be dogmatic about the results, to say some specific test proves, in some way, a specific vendor, platform, protocol, or implementation, is "better." When evaluating tests in the real world, however, we need to be cautious of such statements, and try to examine the entire environment, considering test results with skepticism, and try to understand their limits—as well as their results.

## For Further Reading


[1] V. Manral, R.White, A. Shaikh, "Benchmarking Basic OSPF Single Router Control Plane Convergence," RFC 4061, April 2005.

[2] V. Manral, R. White, A. Shaikh, "OSPF Benchmarking Terminology and Concepts," RFC 4062, April 2005

[3] V. Manral, R. White, A. Shaikh, "Considerations When Using Basic OSPF Convergence Benchmarks," RFC 4063, April 2005.



RUSS WHITE works for Cisco Systems in the Routing Protocols Deployment and Architecture (DNA) team in Research Triangle Park, North Carolina. He has worked in the Cisco Technical Assistance Center (TAC) and Escalation Team in the past, has coauthored several books on routing protocols, including *Advanced IP Network Design, IS–IS for IP Networks,* and *Inside Cisco IOS Software Architecture.* He is currently in the process of publishing a book on BGP deployment, and is the co-chair of the Routing Protocols Security Working Group within the IETF. E-mail: **riw@cisco.com**

# Book Review

## Running IPv6

*Running IPv6,* by Iljitsch van Beijnum, ISBN 1-59059-527-0, Apress, 2005. **http://www.apress.com/**

I've read a lot of books about emerging standards that read like "How I spent my summer vacation at a Standards Body." *Running IPv6* is *not* one of those. While van Iljitsch van Beijnum has been an active part of the IPv6 standards community, he has clearly done the homework of making it all work together. Weighing in at a compact 265 pages, *Running IPv6* really gets right to the point. The reader is assumed to have a working knowledge of IPv4.

### Organization

The book starts off with a fairly typical introduction that explains why the author believes IPv6 is necessary. I find such introductions tedious, because if you've already forked out US $44.95 for the book, the chances are that you're already motivated enough. This is, however, the only tedious chapter in the book.

What follows is a well written and organized primer for network administrators that covers how to configure end hosts, how get address space allocated, set up tunnels, and configure routers and the *Domain Name System* (DNS). The author covers in detail Linux, Windows, MacOS, Cisco's IOS (as well as that of other routing vendors), and Bind. We next move on to applications, IPv6 internals, transition strategies, and transit services.

Throughout, van Beijnum provides practical tips and advice on some of the pitfalls he found so the reader can avoid them. I particularly liked one case of whether to use eui-64 for the lower 64 bits of the address, pointing out the conflict between reducing configuration information (a good thing) and reduced readability (a bad thing).

The book primarily highlights differences between IPv4 and IPv6. This is important because it helps competent IPv4 administrators build on their existing knowledge. I know the last thing I want read about is how routing works when routing itself hasn't changed between versions. And I enjoyed reading, for instance, how *Dynamic Host Configuration Protocol Version 6* (DHCPv6) and stateless address auto-configuration differ from DHCPv4. I did not need nor want a primer in DHCP, but I did want to know about prefix delegation, which is not present in DHCPv4.

The author wastes no time on fluffy protocol niceties. Who cares, for instance, *how* a flow identifier is selected? What's important is that firewalls of the future may take advantage of it to determine flow direction, a major advance. Packet formats and semantics are only provided as they are needed by engineers to determine whether each component is performing correctly. The book is perhaps, therefore, best commended for what it lacks.

Unfortunately it lacks some subject matter I would like to have seen. Although van Beijnum covers how some common user applications, such as *telnet, ftp,* Web browsers and servers, and media players can use IPv6, business applications folks will be disappointed as there is no discussion of Oracle, SAP, or the like. The same is true for network management applications. And this may be a key roadblock to deployment of IPv6, as no self-respecting IT manager would deploy a service that cannot be managed. Such an obvious absence begs the question of whether those applications are IPv6 capable. On the bright side, you can try just about everything mentioned in the book because just about every tool mentioned either comes with the operating system or is freely available on the Internet. This book is not just theory.

### A Must Read

It therefore shouldn't surprise anyone that I consider *Running IPv6* a "must read" for network engineers who have not yet played with IPv6. Even though Network Management Systems and business applications aren't covered, necessary protocol internals, semantics, operations, and troubleshooting are covered, therefore giving the reader a good knowledge base.

*—Eliot Lear, Cisco Systems, Inc.*
**lear@cisco.com**

---

### Read Any Good Books Lately?

Then why not share your thoughts with the readers of IPJ? We accept reviews of new titles, as well as some of the "networking classics." In some cases, we may be able to get a publisher to send you a book for review if you don't have access to it. Contact us at **ipj@cisco.com** for more information.

# Letters to the Editor

## A Pragmatic Report on IPv4 Address Space Consumption

Ole,

Thanks for a great round up in IPJ Volume 8, No. 3 on IPv6. This really helps focus where the state of the discussion needs to be in terms of addressing IPv6 deployment. You might be interested to know that this edition of the IPJ received tremendous interest in the UK. Within 24 hours of it arriving on your website, it was being distributed widely by several mailing lists serving communities from the *Ministry of Defence* (MoD) to important communications industry membership organisations. I received it myself at least three times from different lists!

Over recent months, I've seen a continuing trend to try to sideline IPv6 as not relevant to a particular discussion. IPv6 is either too low level for applications providers to think about, or too far off, or doesn't support some essential infrastructure service today. Some communities feel they have more than adequate IPv4 addresses to meet their foreseeable needs. These factors continue to drive debate on "if ever" rather than on "when" and "how" to deploy. That is, if the debate happens at all. All those who are investing in future IP-related services and networks need to read this edition of the *Internet Protocol Journal* for a reality check.

Tony Hain's article provides a compelling addition to the work you've already published by Geoff Huston on the analysis of IP address allocation, and is important food for thought that I think justifies increasing the urgency with which IPv6 support is treated. The discussion you hosted between Tony, Geoff with John Klensin and Fred Baker I think dealt very clearly with why the debate needs to be focused on the *how* and the *when* rather than on the *if*.

In the UK, we are seeing some significant investments made to enable IP level infrastructure with the intent of delivering profoundly new services into the twenty-first century, but none of these major investments appears to have included a vision for IPv6. So I think the point that was made concerning the current failure in making like-for-like investment decisions between v4 and v6 is hugely important for Chief Information Officers and Chief Financial Officers to take to their boards, or we will continue to find people investing for the past, rather than as they apparently believe, their future.

—*Christian de Larrinaga*
**cdel@firsthand.net**

Ole,

The analysis undertaken by Tony Hain and debated by some recognised experts makes it abundantly clear that the deployment of IPv6 is an immediate natural growth path to sustainability and global mass-market penetration of the Internet, beyond its worldwide current rate of less than 15%.

Tony has presented his study in the recent *IPv6 Forum Summits* (Seoul, Taipei, San Jose and Canberra) and obviously took a lot of people by surprise as previous studies maintained the suspense that the deployment of IPv6 should be an incremental transition and not an imminent and real migration. It was therefore decided to responsibly and morally act on this and renew a global Call to Action to set 2008 as a milestone of inevitable smooth transition in a softer form as a Y2K or Yv4 (The Year when IPv4 addresses will become hard to get) and get engineers to plan for it.

A global worldwide press release was published October 11, 2005 and can be read on the web site of the IPv6 Forum:
**http://www.ipv6forum.org**

The IPv6 Forum would like to recognise the work of *The Internet Protocol Journal* in watching diligently this space for the past couple of years and for initiating and orchestrating the constructive and consensual debate included at the end of the study, a contribution we trust is of great significance to the global good of the Internet.

*—Latif Ladid, IPv6 Forum President*
**latif.ladid@village.uunet.lu**

*The Internet Protocol Journal is published quarterly by the Chief Technology Office, Cisco Systems, Inc.*
*www.cisco.com*
*Tel: +1 408 526-4000*
*E-mail: ipj@cisco.com*






The Internet Protocol Journal, Cisco Systems
170 West Tasman Drive, M/S SJ-7/3
San Jose, CA 95134-1706
USA

ADDRESS SERVICE REQUESTED

PRSRT STD
U.S. Postage
**PAID**
**PERMIT No. 5187**
**SAN JOSE, CA**