IEEE Internet Computing

www.computer.org/internet computing

An Introduction to Internet Standards

Barry Leiba

Vol. 12, No. 1
January/February, 2008

IEEE computer society

An Introduction to Internet Standards

Barry Leiba • *IBM T.J. Watson Research Center*

Internet standards allow hardware and software from different sources to interoperate, and we can do virtually nothing on the internet without them. Here, Standards' new department editor discusses Internet standards in general, giving a brief overview of why they're necessary. He then delves into specific standards for email that aim to reduce spammers' ability to lie about who sent an email message (spoofing) — the Sender Policy Framework, Sender ID, and Domain Keys Identified Mail.

The Standards department has been homeless for the past few months or so. It's settling down now, and, as its new editor, I'd like to use this installment to introduce myself and talk a bit about Internet standards in general, and about this department in particular.

I'm a software researcher at IBM, and I've been working – on and off – on email and email-related projects since the early 1980s. I became involved in Internet standards in the early 1990s, mostly through the IETF, and I still work primarily with that organization. This past March, I was appointed to the IETF's Internet Architecture Board. I have a strong connection to Internet standards and firmly believe that the Internet will be stronger with better, more robust standards.

Why We Need Standards

When I explain Internet standards to people who aren't aware of what they are or why we need them, I usually start by talking about a toaster. Toasters come in various sizes and styles, and they operate in different ways – some have slots that you put the bread into, and the toast "pops up" when it's done; some move the bread past the heat; some just have a stationary shelf for the bread – but they all have some things in common. They perform the same basic function: toasting bread. They also all operate using the same power supply and connect to it with a standard plug (at least within a given country). Because of those two common aspects, you can buy any toaster from any manufacturer, and it will plug into your wall socket and toast your bread. From there, you decide which toaster you like better, without artificial limitations (you don't have to buy the toaster from the same company that sold you your house, for example).

The same is true, of course, with a computer. You plug it into a standard power outlet, and any computer you might buy will perform the same basic functions. But in order for those functions to work, many other standards must be involved.

Consider a visit to a Web site: suppose you want to go to http://ieee.org with your browser. At the very basic level, your computer will communicate over a network, so we'll start with the physical network connection. If we're connected by wire, the plugs, wires, and pin connections are covered by ANSI standards that have been extended to international standards, with names like 8P8C (also referred to as RJ45), T568B, and Category 6. We don't usually think about the standards at that layer, any more than we think about the power cords on our toasters or computers. Next, we look at the LAN standards, for which we almost always use Ethernet these days – an IEEE standard, defined as 802.3, and, in fact, several IEEE 802-series standards might be involved here.

The Web browser, of course, knows none of this; rather, it talks to the operating system's

1089-7801/08/$25.00 © 2008 IEEE

```
220 mail.isp.example Welcome to our mail server
HELO spammer.example
250 mail.isp.example Hello spammer.example
MAIL FROM:<carol@dslprovider.example>
250 2.1.0 OK
RCPT TO:<joesmith@cablecompany.example>
250 2.1.5 OK
DATA
354 enter message
From: Customer Service <cust@yourbank.example>
To: Valued Customer
Subject: Your account

There is a problem with your bank account… [etc]
.
250 2.0.0 mail sent
QUIT
221 2.0.0 mail.isp.example closing connection
```

Figure 1. A Simple Mail-Transfer Protocol transaction. Text in green comes from the server, whereas all other text is from the client.

Transmission Control Protocol (TCP) stack, which gets us to the TCP/IP standard, Internet and transport layers that come from the IETF. We now see things like Domain Name Service (DNS), which turns "ieee.org" into a numeric IP address (usually using the User Datagram Protocol [UDP], instead of TCP, as its transport layer), and various routing and addressing standards become involved, until we finally reach the layer at which our Web browser communicates with the IEEE Web server.

For that, we use HTTP, and the Web page that we get back is in HTML and uses Cascading Style Sheets (CSS), now involving standards from the World Wide Web Consortium (W3C). Had we visited a secured Web page, such as a login or payment page, we'd also be looking at Secure Sockets Layer (SSL) or Transport Layer Security (TLS), which will themselves involve cryptographic standards to verify the Web server's identity and give us encrypted communication with it.

Oh, and the text that the Web server sends us is probably represented using ASCII (from the American National Standards Institute [ANSI]) or Unicode (from the Unicode Consortium).

Just retrieving the IEEE homepage involves more Internet standards than we can easily keep track of from a variety of standards organizations. Doing other things, such as reading and sending mail or viewing blog and news feeds, involve still more standards, with acronyms like SMTP, POP, IMAP, NNTP, and RSS. And I haven't even mentioned all the standards organizations yet; there are others, such as ITU-T, OASIS, and OMA.

We can do essentially nothing on the Internet without using Internet standards because those standards allow hardware and software from different sources to interoperate.

In this issue, I'll take the rest of my space to talk about something close to my own heart: email standards that aim to reduce spammers' ability to lie about who sent an email message, a spamming technique called *spoofing*.

Email and Sender-Spoofing

Email, like paper mail, has an "envelope" — the Simple Mail-Transfer Protocol (SMTP)[1] transaction that sends the message on its way. In that transaction, on that envelope, the sender includes his or her email address along with each recipient's. But the sender's email address isn't verified in any way, nor are any email addresses that appear in the message itself[2] or might be displayed to the recipients. To make things worse, the message might include a human-friendly name to go along with the "from" email address, and many email programs will display only this name and not the address.

When we put this all together, it results in numerous ways that senders can lie about their identities, each way having somewhat different effects. The overall effect, though, is that the recipient has no idea whether an email message is really from the party it says it's from. Figure 1 shows an example of an SMTP transaction; text in green comes from the server, whereas all other text is from the client. We'll use this example as we talk further about spoofing and antispoofing mechanisms.

At least three items in Figure 1 have been spoofed. First, the message is actually from someone at a domain called `spammer.example` (although that, in the `HELO` command, can be spoofed too), but the "from" on the envelope is `carol@dslprovider.example`. That's the address delivery-failure messages will go to, and spoofing it insulates the spammer from those inevitable "bounce" messages. The "from" in the message itself is given as `cust@yourbank.example`, an attempt to look like a message from Your Bank. Finally, the "friendly" name associated with that address is `Customer Service`. An email message might also contain other header fields that relate to the message's sender (for example, "sender," "reply-to," and "re-sent-from"), but I won't go into them in detail here. Most email programs will only show the "from" in the message, and many will hide the address and show only the friendly name.

Antispoofing Standards

Antispoofing standards aim to let an email system at the receiving end detect the spoofing of at least one of these addresses, so that the receiving system can use that information to decide how to handle the message. A questionable message might receive closer scrutiny, be moderated by a human, or get discarded altogether, depending on the situation.

Let's look at three standards that aim to combat spoofing:

- Sender Policy Framework (SPF),[3]
- Sender ID, and[4]
- Domain Keys Identified Mail (DKIM).[5]

SPF and Sender ID both work at the perimeter, where one domain's mail server connects to another. They compare the incoming mail server's IP address with information provided by the domain purported to be sending the message, making sure that the mail server in question is authorized by the purported sending domain. The two techniques differ in some details, most notably which "sender" they consider: SPF looks at the SMTP "mail from" domain, whereas Sender ID uses an analysis of several email header fields to choose the domain.[6]

DKIM, in contrast, uses a digital signature — signing the message with a private key the sending domain owns — coupled with published information about the signing practices[7] of the domain in the message's "from" header (the part of the standard involving the publishing of signing practices isn't yet complete). Because IP addresses aren't involved, the signature can be checked at points other than the perimeter, and it might even be checked in the recipient's mail client.

A Few Standards Organizations

- American National Standards Institute (ANSI); www.ansi.org
- Institute for Electrical and Electronics Engineers (IEEE); www.ieee.org
- International Telecommunications Union (ITU); www.itu.int
- Internet Engineering Task Force (IETF); www.ietf.org
- Open Mobile Alliance (OMA); www.openmobilealliance.org
- Organization for the Advancement of Structured Information Standards (OASIS); www.oasis-open.org
- World Wide Web Consortium (W3C); www.w3.org

What the Standards Do

It's worth highlighting here that these standards are meant for one purpose only: they allow a receiving domain to detect some form of spoofing. Each of these standards will indicate to the receiving domain whether the "sender" identity that they're protecting — which differs among the three techniques — has been verified. The receiving domain decides what to do with the information.

What the Standards Don't Do

These standards receive frequent criticisms, most of which come from misunderstandings about what they don't do. Let's look at some of those criticisms here.

SPF/Sender ID/DKIM won't stop spam. That's correct, and they're not designed to. There's a wide misconception, often promulgated by ill-informed news items, that these are antispam techniques. To the extent that a receiving domain considers spoofed mail to be spam, we might consider these to be techniques to identify spam, yes, but the distinction is important: these techniques are designed to identify mail for which the sending domain (as defined by each technique) is confirmed.

Most mail that passes SPF checks is actually spam! This statement, or some variant ("most domains that publish SPF records are spam domains," or whatever), is commonly presented as a criticism, but considering it so misses the point of these standards. Because they confirm the sending domain's identity, spammers' use of them simply confirms the spam source's identity. Couple that with some knowledge of which domains do and don't send spam — a reputation service, for example — and we have quite valuable information. Far from being a weakness in the system, this is doing exactly what it's intended to do. We would love to have all the spammers admit to who they are, to have them sign all their mail, to have them use known, verified domains — finding spam would clearly be far easier if they did.

This doesn't help for mail from botnets. This point is quite true. Botnets, or *zombie networks* — networks of end-user computers that have been compromised by malware — create one of the most difficult challenges in the antispam fight, defeating antispam techniques such as rate limiting and block-listing.

When zombie computers try to send mail directly into the Internet, these antispoofing standards will, in fact, help — SPF and Sender ID will detect an unauthorized IP address, and the messages won't have valid DKIM signatures. But today, most zombie networks are set up to use their normal email infrastructures to get the mail out, giving that mail the legitimacy of that infrastructure. Judicious use of client authentication can help reduce zombie software's opportunity to send mail, even in this case.

Nevertheless, antispoofing techniques have value even around botnets. A zombie in the `example.com` domain must admit to being in that domain to pass the tests; it can't spoof well-known domains, such as banks and credit-card companies, which, in itself, is important. As with the previous item, we benefit from making spammers admit where their messages are coming from.

This will break the open email system; domains will delete legitimate mail! The standards recommend against routinely deleting unverified mail simply on the grounds that it failed verification. That said, some receiving domains will delete it anyway, despite any suggestions that it's an unwise policy. The antispam community has, in fact, seen domains that do this.

I can argue, however, that antispoofing standards don't make this situation worse. Domains that make these decisions are adopting unusually strong policies — some might say draconian ones — against spam, and they often have very strict spam filters that would cause a lot of legitimate mail to be marked as spam and deleted anyway. The antispoofing standards give them an opportunity to whitelist verified mail from known "good" domains, reducing the risk that mail from those domains, at least, will be flagged.

The State of Things

After discussing the pros and cons of SPF and Sender ID, the standards community couldn't come to consensus on one merged IP-based technique, and so both still exist. Each has been published through the IETF as an *experimental standard*, and the situation will likely be revisited in the future.

The IETF published DKIM in May as a *proposed standard*, the first step on the IETF's standards track. The DKIM working group is encouraging new DKIM implementations and is recommending that implementers of the older Domain Keys mechanism (now published as *historical*) switch to the DKIM standard.

The DKIM working group is also still developing the DKIM Sender Signing Practices (SSP) standard, which will probably become a proposed standard in mid-2008. This is an additional, optional DKIM feature that lets mail-sending domains publish information about how they use DKIM, and thus allows receiving domains to use that information in deciding how to handle mail that purports to be from a particular sending domain but doesn't have a matching DKIM signature.

Over the coming issues, I hope to present a variety of standards from different standards organizations. We'll look at new standards that are coming and old ones that are evolving. We'll discuss interoperability problems and how they're being addressed. We'll consider some gaps in existing standards and see proposals to fill them. I hope the result will be interesting and educational, and that you'll enjoy your visits here.

References

1. J. Klensin, ed., "Simple Mail Transfer Protocol," IETF RFC 2821, Apr. 2001; www.ietf.org/rfc/rfc2821.txt.
2. P. Resnick, ed., "Internet Message Format," IETF RFC 2822, Apr. 2001; www.ietf.org/rfc/rfc2822.txt.
3. M. Wong and W. Schlitt, "Sender Policy Framework (SPF) for Authorizing Use of Domains in E-MAIL, version 1," IETF RFC 4408, Apr. 2006; www.ietf.org/rfc/rfc4408.txt.
4. J. Lyon and M. Wong, "Sender ID: Authenticating E-Mail," IETF RFC 4406, Apr. 2006; www.ietf.org/rfc/rfc4406.txt.
5. E. Allman et al., "DomainKeys Identified Mail (DKIM)," IETF RFC 4871, May 2007; www.ietf.org/rfc/rfc4871.txt.
6. J. Lyon, "Purported Responsible Address in E-Mail Messages," IETF RFC 4407, Apr. 2006; www.ietf.org/rfc/rfc4407.txt.
7. E. Allman, M. Delany, and J. Fenton, "DKIM Sender Signing Practices," IETF Internet draft, work in progress, Nov. 2007; http://tools.ietf.org/wg/dkim/.

Barry Leiba is a senior technical staff member at the IBM T.J. Watson Research Center. His research interests include Internet messaging (including email, instant messaging, voice over IP, and related protocols), abuse prevention (such as antispam and antiphishing techniques), and computer, network, and information security. Contact him at leiba@watson.ibm.com.