HOBBIT

DNS-based spam rejection

Hobbit has played the "Internet insecurity" game long enough to realize that it's hopeless as long as those bothersome and complacent humans are still in the loop. When he isn't working on heavy-handed approaches to spam control, he's probably outside hacking on the Prius.

hobbit@avian.org

TIRED OF BEING HAMMERED BY spam and virus attacks from spyware-infested ISP customers? Here is an easy and entertaining way to use the providers' own DNS naming schemes against them. In the absence of ISPs really doing anything on their end, we can use a few well-constructed filtering rules at our end to deny email delivery from suspect networks, and still leave a path open for legitimate messages.

It is no secret that a huge flood of spam and malicious email emerges from compromised machines in homes and businesses. The botnets grow larger by the day and have become a profitable underground offering. 0wned machines are used to launder connections and launch all sorts of attacks, using the convenient high-speed bandwidth in the customer infrastructure of the ISPs that connect them. Unfortunately, many major ISPs are behind the curve in preventing this, and obstinately remain so even though what they *should* be doing is common knowledge. They want to retain their neutral status as a carrier, not to be responsible for traffic filtering.

Many ISPs have been forced to block downstream packets to obvious problems such as Windows file-sharing ports, but they have put little effort into upstream filtering, thinking that it would cause too many (more) support complaints. Some, such as UUnet and Concentric, have denied direct SMTP delivery from their own untrusted customer clouds, and they've been successfully running such configurations for years. The result is that spam and virus/trojan attempts rarely, if ever, arrive from their infrastructures. I believe that Comcast began experimenting with filtering in the cable-modem swamps, but that seems to have been undone recently.

What is one to do about the rest of the ISPs that simply let the stuff out? Well, another area that ISPs do seem to pay more attention to is DNS naming within the customer networks, and it turns out we can rely on that much more than their ability to keep a lid on the botnets. Largely automated within turn-key DHCP servers, each address that appears on cable and DSL networks generally receives some kind of valid PTR record in the DNS server authoritative for those .IN-ADDR.ARPA blocks. And since the naming is machine-generated, based on the client's IP or MAC address, it usually follows some recognizable pattern such as these:

```
c-24-128-171-15.hsd1.ma.comcast.net
pcp05184511pcs.plsntv01.nj.comcast.net
```

```
pool-151-203-213-167.bos.east.verizon.net
fl-71-0-153-49.dyn.sprint-hsd.net
15-95.200-68.tampabay.res.rr.com
dsl-67-114-79-114.dsl.lsan03.pacbell.net
CPE0008a10ba047-CM014100000470.cpe.net.cable.rogers.com
```

Almost all of these names imply something about the client and/or the surrounding network infrastructure. They are unlikely to be applied to infrastructure machines for the ISP itself, such as the mail-relay servers customers are expected to use for their outbound mail. Despite the fact that DNS is considered a generally untrustworthy source of data, these PTR names are at least somewhat trustable, since an SMTP server will generally look them up "out of band" via servers that are less likely to be under a spammer's control. If a spammer is intercepting all of your DNS queries, then you've got a much larger problem.

Thus, as *part of* an overall best-practice set of anti-spam measures, we can take advantage of such naming schemes and detect if a generic customer is attempting to deliver mail directly, instead of passing it through the local ISP. We can then deny delivery with a rejection message asking the sender (if it's a human paying attention at all) to please use the ISP's authorized relayer to send the message. I do this within Postfix, and all examples herein are based on the filtering features that Postfix has to offer, but the concepts should easily map to other common SMTP servers. We can hope that the ISP mail relay also applies a few anti-spoofing rules to make sure the headers aren't forged—this is in fact probably not the case in most instances due to the management overhead, but as luck would have it, the mechanism works because most ISP customers get configured by default to drop off their mail at the ISP relay. The few who want to do their own direct sending will see the error and, hopefully, resend the message via the ISP to get it through. (Mildly political rant, below, about the relative merits of each path.)

So, how is this implemented? In Postfix, the SMTP server is able to look up and act upon details about the client connecting to it. Restrictions may be imposed by adding one or more instructions to check client name or IP attributes in main.cf:

```
smtpd_xxx_restrictions =
  ...,
  check_client_access regexp:/etc/postfix/client_acl,
  ...
```

where xxx can indicate one of several stages of SMTP delivery. The most common section is smtpd_recipient_restrictions, which allows collecting as much log detail about the client and the message as possible. The regexp: tag can also be hash: or dbm: or pcre:, depending on which style of data storage is desired and which version of Postfix, but it is likely that only regular-expression-based matching will be useful here. The client access directive is usually found somewhere in a list amidst several other types of checks and special cases—allowing local or authenticated connections, looking up addresses in RBL services, etc.

The client_acl file itself contains an ordered list of expressions to match, actions to take upon match, and optional error-message text for rejections. Here's where the DNS-based magic happens, along with any other network-level checks needed. Rules are matched for both DNS lookups and the ASCII representation of the IP address, which rocks because of the seamless versatility that it lends. Some examples:

```
# *all* of cybermall, 207.0.62.0/23
/^207\.0\.6[23]\./                    550 No Soliciting
# oops, several of yahoo's legit relays keep landing in spamcop
+^216\.155\.201\.[56]+                OK
# block all of Latvia [I don't know anyone there]
/\.lv$/                               550 Denied
```

```
# random known spam-nests
/cablemas\.net$/                      REJECT
/\.popsite\.net$/                     550 Access denied (spam)
```

Note that it's still all regexp string comparisons, so we can't specify CIDR notation, but that's fine—to deny partial blocks or aggregates, the [set] syntax of address digits gets close enough. I tend to err on the broader side if I'm slamming the door on some podunk ISP that appears to be spammer-friendly. Postfix regular expressions have a couple of very minor differences from normal ones, and by default are case-insensitive unless a modifier is used. Rule processing is normally "at first match, take the action and exit the list."

Inside the set of client rules, two approaches can be taken for filtering a given provider: either whitelisting its legitimate relay servers and denying the rest, or blacklisting its known customer networks using patterns. The approach taken often depends upon how the ISP does its naming, where it locates its mailservers, etc. It takes a bit of digging and actually *reading* the spam carefully to get a clear picture of their infrastructure, but once that's done the rest is easy.

More examples—first, of names that lend themselves to easy one-shot identification:

```
# general classes of dialup/DHCP clients
/[-.]dial/  550 Use your ISP's mail relay
/dial[-inu.]/                         550 Use your ISP's mail relay
/dhcp[-0-9]/                          550 Use your ISP's mail relay
# AOL direct-dialup customers get dropped into this swamp
/ipt\.aol\.com$/                      550 Use your ISP's mail relay
/ipt\.aol\.net$/                      550 Use your ISP's mail relay
# ATTBI dynamics—client-mumbledyfoo.attbi.com
/client.*\.attbi\.com$/               550 Use your ISP's mail relay
# typical comcast client naming
/client.*\.comcast\.net$/             550 Use your ISP's mail relay
/^pc.*\.comcast\.net$/                550 Use your ISP's mail relay
/\.hsd[123]..*\.comcast\.net$/        550 Use your ISP's mail relay
```

An example with special-casing—I have some correspondents in the Albany, NY area who usually send directly, so I let them in ahead of the rest of Verizon's dynamic blocks. Even trying to keep it this tight still lets spammers connect once in a while:

```
/pool-129.*\.alb\.east\.verizon\./    OK
/pool-141.*\.alb\.east\.verizon\./    OK
# all dhcp/pppoe verizon clients seem to match this...
/pool.*verizon\.net$/                 550 Use your ISP's mail relay
```

For Roadrunner, looking up their MXes gives a pretty clear picture of which naming classes are likely to deliver legitimate mail—usually from something.mgw.rr.com, but over time I discovered some additional ones:

```
/mgw\.rr\.com$/                       OK
/smtp-.*\.rr\.com$/                   OK
/mx[-0123].*\.rr\.com$/               OK
/\.rr\.com$/                          550 Use your ISP's mail relay
```

An example done purely by IP address—a particular business I was dealing with is unfortunately located within one of Electric Lightwave's netblocks. ELI appears to be a spammer petri dish, so they don't get to talk to me, period. Rather than ask the business to change to a better provider, we can allow mail from a small chunk they're in and then deny the rest of their main /15:

```
## SPL-case: SR-systems sits within an ELI block, but let 'em in
/^208\.187\.213\./                    OK
/^208.18[67]\./                       550 Denied (ELI, spam)
```

Obviously, techniques like this aren't for everyone and are certainly not a solution by themselves, but they do kill quite a bit of spam once the major botnet offenders are added in correctly. It is prudent to research as much as is externally visible about each ISP—using spam we receive through them, legitimate messages we receive, MX lookups, their customer-service Web pages, and possibly even Googling for archived message headers in postings from some of their customers just to read how their mail was delivered. For large IP blocks it may be better to use no-logging firewall rules to completely hide the mailservers from them and avoid piling up big log files of rejections. As jingoistic as it may seem, I have had to occasionally deny huge RIPE/APNIC/LACNIC allocations right up front just to stem the tide.

There are also many other techniques that can be done inside Postfix that are beyond the scope of this article. There are plenty of FAQs on spam-busting kicking around and pointed to from the Postfix.org Web site. A nice feature of using Postfix's own features, such as client parsing, is that it's fast—the regular expression matching runs natively right inside Postfix and doesn't need to farm out to external plug-ins or start up extra processes and Perl interpreters. SpamAssassin and the like seem somewhat less attractive because they sit outside Postfix and chew much more CPU. But if the external packages have the other features you need, then by all means use them.

Some ISP customers, such as small businesses and consultants, may have a legitimate need to handle their own mail and deliver directly. Unfortunately, the ISPs often tar them with the same dynamic-DNS brush, so when the business believes it's acewidgets.com, the outside world sees its SMTP connections coming from c-67-163-134-87.hsd1.ct.comcast.net. The right answer for this is either delegated reverse DNS as described in RFC 2317, or for the ISP to maintain a nominal set of static PTR records for that customer. However, finding anyone within the ISPs who even knows what that means, let alone how to set up and maintain it, is often an insurmountable challenge.

Forcing customers through ISP mailservers is also a political gray area, as well as a certain amount of maintenance headache for the ISP. It doesn't have to be. The relay servers are usually already in place, since most customers appear to get set up to deliver through them by default. But those servers also present a good opportunity for the ISP to enforce anti-spoofing in message envelopes and headers and be a better neighbor to the rest of the Internet, especially with today's nonstop epidemic of forged headers and joe-jobs coming from compromised customer machines. It is highly worthwhile for them to build infrastructure that is sufficiently capable and configurable to support that enforcement but still handle some special cases, such as letting acewidgets.com deliver as acewidgets.com, and then lock down the customer subnets. This of course requires having those same people who are clueless about delegated DNS come to understand how to run a proper mailserver.

If I seem to be down on ISP technical capability in general, it's for very sound historical reasons. Really, I have tried to get through to many of them and have consistently failed to find anyone who knew what I was talking about. Perhaps if you're reading this and work for one of these ISPs, you can take it to management as a wake-up call. The more ISPs realize the extraordinary extent of the problems and put forth the effort to be better Internet neighbors, the less need there will be for kludges like all of the above.

REFERENCES

"Classless IN-ADDR.ARPA delegation": http://rfc-editor.org/rfc/rfc2317.txt

Postfix FAQs: http://www.postfix.org/docs.html