Fine Grain Spam Suppression Using Reachback

Dennis Heimbigner

CU-CS-1016-06

November 21, 2006

University of Colorado at Boulder
Technical Report CU-CS-1016-06

Department of Computer Science
430 UCB
University of Colorado
Boulder, Colorado 80309-0430

Fine Grain Spam Suppression Using Reachback

Dennis Heimbigner

Computer Science Department
University of Colorado
Boulder, CO 80309-0430, USA
dennis.heimbigner@colorado.edu

***Abstract**: A system is proposed that is capable of identifying the true originator of an email message. This information can be combined with existing whitelist and blacklist technology to provide an effective anti-spam system. The approach is to store validation information that can be traced to the sender and to utilize a "Reachback URL" in the message to access the validation information. The approach proposed here has a number of advantages over existing systems. It does not require use of DNS, it can use the existing email distribution system, it is robust against in-transit header modifications and mail forwarding, and it can identify sources down to the granularity of individual email addresses. It can be incrementally deployed by individual users and can provide incremental value through automated whitelisting.*

1. INTRODUCTION

Spam email (spam for short) is an unsolicited email message that has been sent indiscriminately to massive numbers of users. Despite major efforts to suppress spam email, the distribution of spam continues to plague email users. Estimates vary, but there is some consensus that 60-80% of all email is now spam email. The problem has become especially pressing in recent months [15] because sites have seen a 50% increase in spam. This represents a tremendous burden on users.

To date, there are three primary approaches for suppressing spam. Content filters such as SpamAssassin [2] are the most common approach. A content filter examines each email message and based on its content, the filter provides an estimate of the probability that that piece of email is spam.

Another approach is called "whitelisting" [13]. A whitelist is a set of addresses that the receiver believes to be non-spam sources. Any email from a site on the whitelist is accepted and all others are rejected.

The other common approach is called "blacklisting" [1,9,18,19], which is essentially the complement of whitelisting. A blacklist is a set of email addresses that are believed to be sources of spam email.

Blacklists and whitelists can be used in combination. As a rule, a whitelist for an individual user will be much smaller than a blacklist. So it is useful to test the email address first against a whitelist. If it is on the whitelist, then there is no need to perform the more costly blacklist check.

Blacklisting and whitelisting depend critically on the ability to accurately identify the true source of an email message. Unfortunately, it is easy to forge ("spoof") the source address in the *From* header in an email message [6]. This means that a spammer can send email that has a fake source address that will not be caught.

This paper proposes a novel mechanism for accurately identifying the true source of an email message. Briefly, it adds information to each email message that provides access to a source of validation information that produces unforgeable evidence of the true source of the email address. Generically, this new method (and associated infrastructure) is referred to as "Reachback." The Reachback system computes verification information about each sent email. The original message is augmented with a special "Reachback" header containing a URL (Uniform Resource Locator) [5]. This Reachback URL points to an information source – a Reachback server – that contains the information needed to verify a specific email message. This combination of server, of its internet IP address, and of the validation information combination allows a recipient to reliably infer a connection from the email message to the Reachback URL, then to the Reachback server, and finally to the email sender.

Reachback has several advantages compared to existing approaches.

- It does not use the Domain Name System (DNS) to store validation information.
- It can be used both for blacklisting and as an automated whitelist.
- It can utilize a variety of servers to provide Reachback validation information; it can even delegate the validation information to servers outside of the sender's domain.
- It can support very fine grain validation: down to the level of individual email addresses.
- It has the option to use simple secret keys rather than public/private pairs because the decryption burden can be transferred to the Reachback server.

2. SPAM SUPPRESSION USING REACHBACK

Spam suppression is initiated by the receiver when the email is received by the mail transfer agent (MTA) for an intended recipient of the message. This email is in the form of a normal

email using the normal email transport mechanisms. Note that the message may have traversed any number of sites before arriving at the recipient's MTA. This means that it may have been forwarded, passed through a mail list, or even passed through an open email relay.

Either the recipient's mail transfer agent (MTA) or the recipients email client program (aka mail user agent (MUA)) carries out the following procedure to validate a received email message.

1. Extract the Reachback URL from the appropriate email header line.
2. Use the URL to obtain the verification information stored at the Reachback server at the source.
3. Apply the verification information to the message to accurately validate the source of the email message. The Reachback URL header line is left in the validated message but rewritten to indicate that it has been used to validate the content.
4. If the validation fails, then the email is marked as not invalid.
5. If the validation succeeds, then the URL is tested first against a whitelist of known good sources, and then against a blacklist to determine if this particular source is a known spam source. The message can then be marked with that determination.

3. REACHBACK FOR WHITELISTS

In order for Reachback to achieve wide-spread use, there must be some advantage in incremental adoption. Reachback provides this advantage in the form of an effective automated whitelist in addition to it use for blacklists. Any clique of users that mutually adopts Reachback is immediately guaranteed that the members of the clique are not spammers (or will be suppressed if they are). The advantage over traditional whitelists is that the set of acceptable senders can grow without action by the previous members of the clique. This can be a big advantage in certain institutions such as Universities. The clique is the set of personnel in the University, which is a rapidly changing population. Reachback implementation is also relatively easy because email transmission is often a centralized function in Universities and because most University personnel have access to a personal web page. Every user that adopts Reachback automatically enters the clique of accepted users, and this provides an incentive to adopt Reachback.

4. REACHBACK VALIDATION INFORMATION

The key to the operation of Reachback is the validation information returned by accessing a Reachback URL. This information consists of the following items.

1. Key: A public key from a public/private key pair.
2. Address: An email address.
3. Signature: A signature covering the information in items 1 and 2 and computed using the private key associated with item 1.

For convenience the term RVI is used for combination of three items and the three items are referred to by *RVI.key*, *RVI.address* and *RVI.signature* respectively.

5. REACHBACK URL FORMAT

The form of the Reachback URL is critical for inferring the source of an email. Recall that a URL is of the general form "http://domainname.tld/path" [5], where "domainname.tld" is the DNS name of a particular machine and "path" is a sequence of names separated by the forward slash character.

For a given domain name prefix, the set of possible paths effectively forms a tree. Each specific path can be used to identify an individual source. Thus the URL for a message from user Bob might map to the URL "http://yahoo.com/Bob." The assumed format is actually a little more complex because it must be possible to algorithmically infer the email address of the sender from the URL; as discussed in the next section, this is necessary for comparison with *RVI.address*.

Consider an email with *RVI.address* equal to

"dennis.heimbigner@cs.colorado.edu."

This must have a Reachback URL of the form

"http://www.cs.colorado.edu/X/dennis.heimbigner,"

where X is some universally standardized file prefix.

This URL structuring makes it possible for blacklist systems to attribute the email source to a very fine degree. So rather than blocking all of Yahoo because Bob is sending spam, only Bob needs to be blocked, and this can be applied by using the Reachback URL.

6. SOURCE VERIFICATION USING REACHBACK

At the sending source, a digital signature for the email is computed using a checksum of the email's content plus a private key from a public/private key pair. This digital signature is included in the email message. For simplicity, it can be assumed that is it included as an otherwise ignored parameter in the Reachback URL, but it could be placed, for example, at the end of the message body.

An email message recipient obtains the RVI using the Reachback URL in the email message. This RVI is itself validated by using the *RVI.key* field information to validate the *RVI.signature*. Given the valid RVI, the email message is validated as follows.

1. Compute the checksum of the message body (call it CSUM).
2. Use *RVI.key* to decrypt the digital signature to produce the sender's checksum. Compare this to CSUM. If they do not match, then declare the message invalid.
3. Verify that *RVI.address* is consistent with the Reachback URL as described in the previous section.
4. If the validation passes steps 2 and 3, it can be inferred that the message did indeed come from the email address specified in *RVI.address.*
5. Test *RVI.address* against a whitelist of accepted sources and/or a blacklist to determine if the message came from a known spam source. Mark the message accordingly.

The Reachback URL defines the true source of an email. Unlike other approaches, Reachback can completely ignore any *From* header in the email. The information accessed through the Reachback URL is sufficient to both identify the source and to verify that the email came from that source.

7. THE REACHBACK SERVER

It must be possible to associate the source of the RVI to the sender of the email. This allows a recipient to infer a connection from the email message to the Reachback URL, then to a Reachback server, and finally to the email sender.

Any server that can be accessed by some well-known URL protocol can be used as the server for Reachback information. The simplest form of information source is an HTTP server that is trusted by the sender and is located in the same domain as the sender. The Reachback information is placed at a well-known location under the sender's web page. The rest of this paper will assume this is the case.

8. COSTS

This approach is not without cost. It requires resources from each site that sends or receives email using Reachback.

1. The sending site must support an HTTP-server for the users at its site. This server is used for accessing validation information about email messages that it distributes.
2. The sending site must store for some period the public keys used to validate the signatures.
3. The sender must bear the cost of computing the digital signature.

4. The receiver must bear the cost of validating the digital signature.
5. The receiver may optionally bear the cost of caching Reachback information (Section 9.3).

None of these costs is especially onerous.

9. IMPLEMENTATION ARCHITECTURE

Implementing Reachback requires the insertion of three new components into the normal email transfer architecture. These components are (1) the HTTP server, (2) the sender-side proxy that inserts the Reachback URL into the email message, and (3) the receiver side validator. There are a number of ways to do this insertion as described in the next two sections.

9.1 Sender-Side Implementation

Figures 1a and 1b show two possible placements for the sender-side proxy. Whatever the placement, the proxy takes a message and constructs the validation information (the digital signature) from the message contents and a private key. The critical assumption is that the sender maintains a web page to hold the corresponding public key needed to validate the digital signature. Further, the server holding this page must be accessible from any potential recipient of email coming from this sender.

Figure 1a is preferred because it can provide a Reachback system for a whole site. However, Figure 1b is valuable if an individual user wants to implement Reachback without waiting for site-wide implementation. This is a significant difference with DNS-based approaches, where site-wide implementation is necessary. Incremental adoption by individual users is effectively impossible when DNS is used.

It is also possible to define some variations on the placements in the figures. For each placement, it is possible to merge the proxy with the component on either side: the email client or the MTA (typically an SMTP server). The latter is preferable because it allows the insertion of a Reachback URL into every message. This includes automatically generated emails indicating error conditions. This is not possible when the proxy is merged with the email client because that client will never see the error email.

The HTTP server is also subject to merging with the MTA. This is possible because the MTA is usually already accessible from elsewhere on the Internet, hence it is easy for it to also export an HTTP server interface. This situation is unlikely to be needed in practice because of the general availability of web servers. It would however be useful if other key signing protocols are used (Section 12).

9.2 Recipient-Side Implementation

Figures 2b and 2b show two possible placements for a receiver-side validation proxy. As with the sender side, this proxy may be merged with components on either side. This proxy takes a message, uses the Reachback URL to obtain the validation information and validates the message.

Figure 2a shows the preferred placement. Here the validation proxy acts as an SMTP proxy that receives all incoming mail, validates it, and passes it on to the site's normal MTA: Postfix, for example.

Figure 2b places the proxy between the usual POP3/IMAP server and the email client. The proxy implements a wrapper for POP3 or IMAP mail delivery systems that can perform validation. Every recipient's email client

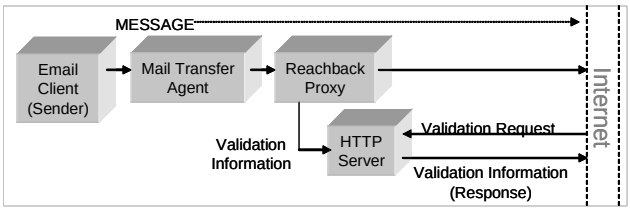

Figure 1a. Sender Proxy Placement: System-Level

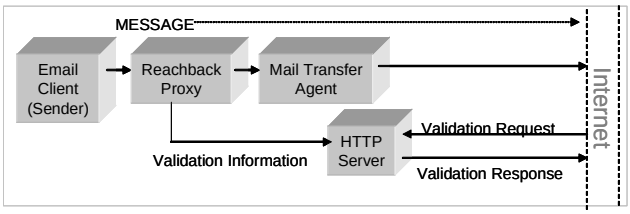

Figure 1b. Sender Proxy Placement: User-Level

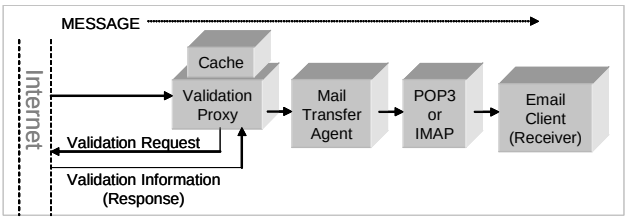

Figure 2a. Receiver-Side Proxy Placement: System-Level

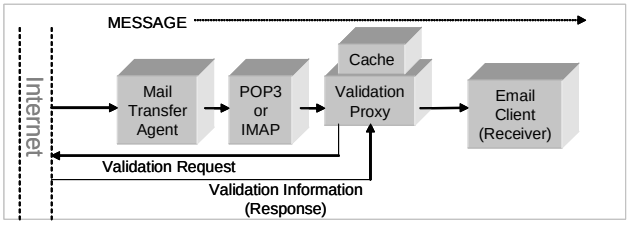

Figure 2b. Receiver-Side Proxy Placement: Userl-Level

program points to the wrapper and the wrapper in turn points to the real POP3 or IMAP server. Commands are transparently passed to the real server, but validation is applied to mail messages. This case is also intended to cover something like the Unix Procmail system, which is per-user.

It is possible to consider placing the proxy between, say, Postfix and POP3, but this is generally difficult because these two programs usually communicate through the file system (using Maildir format, for example) as opposed to using TCP/IP.

Validation before the initial MTA (Figure 2a) is by far the best solution, but it is also the one with the most organizational impact. Figure 2b would be desirable and is feasible for POP3 servers. IMAP servers, however, are so complex that the validation would be difficult to implement. Merging the proxy with the email client again has the advantage of supporting incremental adoption, but is otherwise undesirable purely because is purely a per-user solution.

9.3 Receiver-Side Caching

The receiver-side validation proxy can utilize caching to significantly reduce the cost of validation. Figures 2a and 2b show a cache attached to the validation proxy. The cache maps the Reachback URL to the corresponding public key. Instead of always accessing the sender's HTTP server, the validating proxy first checks the cache and if the Reachback URL is found, it attempts to validate using the matching public key. If the URL is not in the cache, or the cached key does not properly validate the message, then the sender's HTTP server is accessed to obtain the key. That (URL,key) pair is inserted into the cache for subsequent use. Assuming a fixed size cache, some replacement policy must be defined. LRU would be a good choice although there may be semantic knowledge that could produce a better policy. The cache could also maintain a whitelist of senders that should never be replaced.

10. ADDITIONAL ISSUES

There are a number of lesser, but still important issues that must be addressed with respect to the practical use of Reachback.

10.1 Idempotence

Ideally, validation for each message is done once on the sender side and once on the receiving side. In practice, it is likely that some users will implement Reachback themselves. Later, a site-wide deployment may occur. This means that it is possible for an email to have multiple Reachback URLs and/or to pass through multiple validation proxies. It is very desirable, then, if the Reachback process is idempotent so that second or later proxies will properly validate.

On the sender side, repeated Reachback URLs cause no obvious problems. Each additional URL accesses a different public key any one of which is usable for validation, although the initial one is most valuable because it indicates the true sender. One solution is for senders to recognize a message with a Reachback URL header and just pass it on. After all, the original Reachback URL specifies the true and original source for the message. If multiple Reachback URLs exist, then they can, if desired, be validated successively to check all of them.

On the receiving side, the first validation proxy can rewrite the header to flag the fact that it has performed the validation. Subsequent validation

proxies can check for this header flag. This provides a potential attack where the sender formats the message to look like it already has been validated, but still specifying a fake source. The solution is to "trust but verify." Every validation proxy rechecks the validation. This introduces some overhead in the short term until the downstream proxies are removed in favor of the first proxy.

10.2 Forwarding

Forwarding – including various kinds of relays and proxies – involves adding a few new header lines to the email message and re-sending the message to a new destination. This causes problems for competing systems because they utilize the *From* header line to determine which DNS entry to check. In contrast, Reachback does not care about those header lines, which means that it is unaffected by forwarding.

10.3 Mail Lists and Mail Digests

Mail lists operate by collecting email messages from multiple senders and re-sending them to their subscribers. Digests are similar except that they will send out a single message that aggregates a number of submitted emails.

There are four possible combinations of interactions between Reachback and mail lists depending on whether the sender or the mail list site uses Reachback.

1. Sender: no; Mail list: no. There is no effect.
2. Sender: yes; Mail list: no. The messages that are redistributed by the mail list will contain the Reachback URL pointing to the original validation information.
3. Sender: no; Mail list: yes. The redistributed messages will use Reachback and the validation source will be the mail list site.
4. Sender: yes; Mail list: yes. The idempotence argument applies (Section 10.1), but otherwise can be treated the same as case 2.

10.4 Incremental Adoption

It is clear from the experience of the Internet community that adoption of any anti-spam system will be a protracted process. Hence it is important that the system can be incrementally adopted with minimal disruption and with the ability to work with existing email clients and other email infrastructure. If it can be adopted by individual users in advance of site-wide adoption, then that is an important capability.

Incremental adoption of Reachback is possible because individual sites, and even individual users, can begin to adopt Reachback as they see fit. Senders and receivers can adopt Reachback transparently; the only visible sign is the inclusion of the Reachback URL in the set of email headers.

As with all anti-spam solutions, email that does not use the anti-spam solution cannot effectively be tested against a blacklist. Therefore, it is not possible to discriminate against such email until it is adopted widely. For Reachback, this situation occurs for messages that have no associated Reachback URL. In the short term, the only solution to this is to fall back on filtering.

As discussed in Section 3, however, incremental adoption does provides value to users even in the absence of widespread adoption. This is because it supports an automatic whitelisting capability that is useful even in the absence of effective blacklisting.

11. POTENTIAL ATTACK SCENARIOS

It is difficult to guarantee that a given anti-spam system is really secure against attacks. The attackers (spammers in this case) are quite resourceful and may utilize attacks not considered by the developer. This section considers some possible attacks and how they can be addressed.

11.1 Address Spoofing

The primary methods [6] for spoofing email header addresses are zombies, open proxies, open mail relays, and transient internet connections. The last three cause no particular problem because Reachback does not use the *From* header or any other header except the Reachback URL. This means that forwarding

through additional sites has no effect. The zombie issue is, however, of concern and is addressed specifically.

Another possible address spoofing attack available to a spammer is to deliberately use a fake Reachback URL. Obviously using a completely fake URL will fail because no useful validation information can be provided.

11.2 HTTP Server Spoofing

Another kind of spoofing is possible where the spoofer sets up a dummy HTTP server. This server can be used in several ways. First, it can return the same RVI for all Reachback URLs that access it. The spoofer then inserts a fake Reachback URL based on that key and sends out the message. The receiver will then validate the message using that key. This kind of spoofing will not work for very long because that HTTP server site will rapidly be tagged as a spam source.

The spoofer can also attempt to set up a large number of HTTP servers, but as discussed in Section 11.9 this can become prohibitively expensive.

Another possible attack is to set up a server that pretends to provide validation information for a specific sender's email address. This can work only if the spammer is in the same domain as the true sender and has access to the sender's web pages. Serving information from any other site will fail against the URL format validation step. In this situation, the spammer has essentially compromised the whole site, so that spammer must be suppressed by administrative mechanisms.

A less obvious attack is possible where the spoofer uses a special URL with the specific intent to get the recipient's validation proxy to access it. This is a problem because historically some browsers have had serious implementation flaws. These flaws are such that just visiting a certain web site can cause the user's computer to become compromised. Usually this occurs because the web server delivers unexpected content that causes code execution on the client side. This is an argument for doing validation without using any web-browser components. Rather, the validation proxy should be constructed from scratch to enforce the use of a simplified HTTP protocol by the server and thus suppress unexpected content.

11.3 Zombies

A zombie is a machine that has been compromised by some malicious hacker. The zombie's software is modified so that the zombie will execute commands as desired by that hacker. Usually, a zombie belongs to some unsuspecting consumer. That consumer may not have the skills to detect that their machine is a zombie.

Spammers are increasingly making use of zombies to send out their spam messages. The message appears to come from a legitimate source (the zombie machine). The zombie effectively becomes the true source of the message and this makes it hard for all anti-spam systems to suppress such spam. In theory, it is possible to blacklist zombie machines. Unfortunately, it is often the case that the zombie uses some other mail site to actually send its email: Google or Yahoo, for example. Obviously blacklisting everything from Yahoo is not feasible.

Reachback has the potential to help address the zombie problem because it can identify spam sources down to the level of individual users. This means that it is in fact feasible to blacklist those specific users of Yahoo, for example, that are actually zombies and leave other users unaffected. This blacklist information can also be fed back to the mail site to help it identify zombie spammers. No competing approach can do this.

The zombie may use a simple mail relay through its ISP. If the ISP supports Reachback, then the zombie can be properly blacklisted. If the ISP does not, then Reachback cannot say that the email is valid, and it must be dealt with as non-validated mail.

The controller of the zombie may also add a Reachback server to the zombie so that any spam email sent from that zombie (correctly) appears to come from that zombie machine. But recall that validation of source does not mean that the source is not spam. The blacklist determines that property. Thus, a zombie sending validated spam would still end up on the blacklist in short order and all email from it would be suppressed.

11.4 HTTP Server Hijacking

It is possible that a malicious hacker can compromise some other site's HTTP server. This is less likely if the server implements highly restricted functionality. Nevertheless, it must be considered a possibility. In any case, this would appear to be the same problem as server spoofing with the same solutions.

11.5 DNS hijacking

It is theoretically possible for a spammer to hijack legitimate DNS entries to point to one of his machines. This is a difficult attack to execute and is likely to be much more difficult as DNS security improves. This seems to be an issue for any of the anti-spam mechanisms and is not unique to Reachback.

11.6 Receiver Anonymity

The very act of using the Reachback URL to access the sender's HTTP server provides information to the sender. It indicates that the email recipient exists and is reading email and it tells the sender the IP address of the recipient's machine. This may be considered a significant loss of anonymity compared to the current email system. A spammer can use this information to target the recipient with traditional spam. Effective anonymity will return to every user when their site implements Reachback. This is because the validation proxy will be the component that retrieves the content independent of the legitimacy of the recipient addresses. This means the sender knows only the IP address of the machine running the validation proxy. Moreover, the spammer does not learn if any of the recipient addresses are in fact valid.

11.7 Blacklist Poisoning

Standard blacklists are subject to poisoning, which means that fraudulent spam messages are sent with legitimate *From* headers in an attempt to fill the blacklist with legitimate senders, thus rendering it useless. This is much more difficult with Reachback because the maintainer of the blacklist can retrieve the validation information and independently verify that the supposed spam source is in fact the originator of the spam message.

11.8 Massive Email Address Space

Any spammer who has access to a very large number of email addresses can potentially defeat any blacklist system. The spammer need only use each email address in turn to send a large amount of spam. After some period, the spammer moves on to use the next available email address. The obvious solution to this is to blacklist the whole subdomain, or domain, with which all of the email addresses are associated. This works fine when the spammer is using a zombie. It fails if the domain is "special" in that it represents one of the large email providers such as Google or Yahoo. In this case it is impossible to blacklist the whole domain because so many legitimate users would be affected.

Google and Yahoo both allow free email registration, so in theory a spammer could use an automated robot to register as many email addresses as needed. Fortunately, these special domains have recognized this problem and have added mechanisms to their registration process to prevent automated registration. Google, for example, requires the registering person to have a mobile phone address with instant messaging. Yahoo uses a puzzle system that is difficult for automated systems to solve, but is easy for people to solve.

11.9 Massive use of DNS names and IP Addresses.

Massive use of DNS names presents a problem that is analogous to the massive email address problem. At least under IPV4, it is costly to own more than a few IP addresses. It is possible, though, to define an arbitrary number of DNS names as subdomains of some primary domain. In practice, the hierarchical nature of DNS names makes it possible to suppress large number of subdomains by moving up the name hierarchy. This means that while a spammer might have a million names of the form "name.spamdomain.com", they all will share a common suffix: "spamdomain.com" in this case. That primary domain ("spamdomain.com") can then be blacklisted.

11.10 Inaccessible HTTP Server

When a receiver gets an email and wants to validate it, the receiver must contact the specified HTTP server. It may be the case that the server is inaccessible. This may be the case because the server is down or its network connectivity has been severed or it is the subject of a denial-of-service (DOS) attack.

One solution to the problem of server accessibility is to propagate the Reachback information to a number of sites across the Internet. Reachback URL caching (Section 9.3) is one example of this in which the Reachback information is propagated to the receiving site where it may be used even if the sending site is inaccessible.

This approach may be extended by allowing other sites to provide the information and providing multiple, redundant Reachback URLs in the message. This is called Reachback delegation. In order for this work, the receiver must trust that the information at the delegated site is accurate.

A variant of the PKI approach can be used to provide that trust. A well-known and trusted site can provide a service in which it is asked to obtain the Reachback information from some site. The trusted site accesses that information and signs that information plus information about its source. The trusted site uses its own private key, and its corresponding public key is assumed to be well-known. The signed version can then be placed at any convenient location in the web and used as the Reachback URL.

Note that the Reachback delegator only attests to the IP address, DNS name, and Reachback information of the source. This is in contrast to other forms of attestation such as PKI or PGP web of trust, where the goal is to attest to some notion of "identity."

11.11 Active Impersonation

Active impersonation, also called "man-in-the-middle", presents another possible source of attacks. In practice this seems relatively unlikely to be used by spammers because that level of control probably can be used to convert a site to a zombie. Nevertheless, the effects of such an attack are worth examining.

It can be assumed that the active impersonator has the ability to (1) examine and arbitrarily modify any email being sent to a given receiver and/or (2) examine and arbitrarily modify any email being sent from a given sender. It is assumed that in either case, the impersonator has no other control over the sender or receiver.

In either case, it appears that the impersonator has only a limited set of actions that it can take. The impersonator can completely replace the message with one of its own choosing, but that is equivalent to just sending spam. It cannot replace the body of the message without modifying the Reachback URL because the validation would fail. Replacing the URL (URL spoofing) has already been addressed in Section 11.1. The only effective action the impersonator can take is to remove the Reachback URL completely. There is no short-term solution for managing email that has no Reachback URL. Existing filter-based solutions must be relied upon.

12. REACHBACK VARIATIONS

Reachback provides the option of using a single secret key instead of a public/private key pair. Of course, the single key must remain private to

the sender and can never be revealed to any receiver. This method requires the receiver to send a cryptographic message authentication code (MAC) to the source. Operationally, the sender computes the secret key to compute a MAC value for a checksum of the message body. This MAC value is included in the Reachback URL as a parameter. When the sender is contacted using this URL, the sender performs the decryption and returns the unencrypted checksum to the receiver. The receiver compares the unencrypted checksum to a locally re-computed checksum. If they match, then it can be assumed that the email message came from the source specified by the Reachback URL.

The use of a single secret key has one major drawback and two minor ones. The primary reason to avoid it is that local key caching cannot be used, so it must always perform an HTTP communication to the source. Less importantly, a plaintext attach is theoretically possible. An attacker could send an arbitrary value to the sender to be decoded. The returned value would then represent the plaintext for that sent value. This attack is easy to defeat. It is only necessary to force the unencrypted digest into a specific format such as duplicating the digest or adding a constant string to one end or the other. Then if the unencrypted plaintext did not conform to this format, the sender would not return the plaintext, but rather some fixed value indicating failure to decrypt. Periodic rekeying also would aid in defeating this attack.

The other minor issue involves a denial of service attack in which the attacker repeatedly asks the HTTP server to decrypt a signature. This is relatively easy to defeat by introducing an artificial delay into the decrypting process based on the source of the decryption request.

Reachback also offers the possibility of using other validation information in place of keys. Some of these alternatives are defined in the following paragraphs.

1. Email body. The Reachback URL may provide a duplicate of the contents of the email, including selected headers. The content obtained by Reachback can be matched to the email content in the notification. Note that the accuracy of any included headers is irrelevant; it is only important that they match.
2. Replacement. This is a variant of the preceding case. Instead of comparing the contents, the content of the email message is discarded and replaced by the content retrieved through Reachback. This has the advantage that the email message does not actually have to contain the contents at all, which results in smaller messages. This case has two drawbacks. First, such messages cannot be read offline unless validation occurs before being sent to the user. Second, if a non-Reachback user receives such a message, they must invoke a web-browser on the Reachback URL to access the mail's contents.
3. Per-message public key. The URL may provide a public key for the contents of each specific email. This is as opposed to per-user or per-site public keys. The retrieved public key is used to decrypt that specific email message. Of course this approach is not really very feasible if computing a public/private key for each email is costly (as it is).

Note that cases 1 and 3 can also be carried out at the HTTP server. This allows the server to determine validity using any method it chooses and without the knowledge of the receiver.

These alternative validation mechanisms are less desirable than reaching back for a key because they impose a much larger storage burden on the source site's HTTP server, so some form of ageing of per-message validation information must be implemented. Nevertheless, these alternatives may be worth exploring in detail in the future.

These alternatives are not possible with DNS-based approaches because they are inherently oriented to domain level validation and are

limited in the amount of information they can store in DNS.

Reachback is also not restricted in the kind of source server used to obtain validation information. This is because the URL can encode both the protocol and the port to use at the sending site. So ports other than 80 can be used and protocols other than HTTP can be used: FTP for example. The primary requirement for the protocol is that it must support a very large address space capable of encoding some form of standardized URL as described in Section 5. HTTP meets this requirement through the URL structure. FTP can meet it through the use of its file structure.

13. RELATED WORK

Anti-spam mechanism can usefully be classified into several categories: filtering, blacklists, and whitelists, and with variants on each.

13.1 Filters

The most common approach to detecting spam email is to use a filter program that examines each email message for specific features, scores them, and compares the score to a threshold to decide if the email is spam. Bayesian filters are often used to provide the score, but the filter must be trained to separate good email from spam email. When a new email is received, its content is passed through the filter and the estimate is obtained. Based on some threshold, it is then classified as spam or not spam.

Spam filter programs are quite common and are reasonably effective. Users tend to set the spam threshold rather high in order to minimize the number of false-positives; hence they tend to have to deal with a large number of false-negatives. Filter approaches are complementary to all blacklist approaches and indeed are essential for making the initial detection that spam is being issued from some source.

13.2 Whitelisting

A number of whitelist approaches exist. They operate by defining the set of sources allowed to communicate with a given recipient. Two common mechanisms are PGP Mail and Challenge-Response.

PGP Mail [16] uses a private key pair to encrypt email. The receiver uses the private key to decrypt it. If the public key can be definitively associated with the sender, then PGP mail can be used for source verification. In practice, PGP Mail operates as a whitelist. The owner of the public key restricts access to it by giving it to selected trusted receivers; this distribution is equivalent to constructing a whitelist. The goal is to prevent unauthorized reading of the email.

As a rule, each sender and receiver must maintain their own whitelist. This maintenance can be time-consuming since the set of acceptable partners is likely to change fairly rapidly. The whitelisting capabilities of Reachback (Section 3) address this problem since whitelist maintenance is effectively automated.

13.3 Challenge-Response

The challenge-response mechanism [7,12] provides a form of whitelisting. When an email message is received, the receiver returns a separate message to the sender that requires the sending person to solve some form of puzzle. Typically a human is required to solve the puzzle because it is not easily solved by any form of automatic robot. If the action is correctly performed, that source is added to the whitelist. The assumption is that a spammer will not have the resources to respond to such requests and that the requested action is difficult to automate. This approach has not been widely used because it breaks the “fire-and-forget” model for email where the sending user does not need to deal with a sent email until such time as an answer is returned by the sender. Challenge-response requires extra work on the part of the user. So unless the message is rather important, the validating request is just ignored. Further, this approach does not deal well with messages coming from a legitimate mailing list. In contrast, Reachback obtains its validation information automatically whereas Challenge-

Response requires a person to be "in-the-loop" of validation.

13.4 Greylisting

Greylisting [11] provides another whitelisting approach that utilizes the inherent retry capabilities of the email system. Upon initial receipt of an email with a unique combination of sender, receiver, and proximate sending host, it returns a rejection of that email to that host. This rejection is repeated for some short period of time. The assumption is that spam email senders will not attempt to retry the send while legitimate senders will retry. Of course, spammers may adapt to it by retrying sends, but it would appear to be useful as a complement to other spam suppression mechanisms, including Reachback.

13.5 Blacklisting

All blacklisting approaches require the ability to accurately identify the true source of an email. To date, the sources of such information have come from two sources: the *From* header in the email message, or (2) from the remote IP address of the proximate site that sent the email. In the face of header spoofing and email forwarding, the first source is suspect. The second source is reliable but may not be the actual source if email forwarding or relaying has occurred.

In any case, almost all current blacklist approaches utilize the Internet Domain Name System (DNS) to store validation information about the assumed source. The idea is that validation information is inserted into DNS by every legitimate source. The receiver of an email then accesses that validation information and validates the email. If the validation fails, or the sending site is known to be a spam site (based on the blacklist), then the email is marked as spam.

The various DNS-based systems are differentiated by what they use as the sending address and, more importantly, the validation information they place in DNS. All DNS-based systems, however, have certain common drawbacks.

1. Storage Cost: DNS is a shared resource for the whole Internet. Storing validation information in DNS forces all DNS servers to pay the storage cost.
2. Granularity: DNS-based anti-spam cannot support a granularity much finer than an IP address (mail host machine) primarily because that is the level of granularity supported by DNS and also because any attempt to move to a finer grain could rapidly overload DNS servers.
3. Zombies: The limited granularity makes it difficult to do anything about zombies that send email using large sites such as Google or Yahoo.
4. Incremental Adoption: DNS-based systems require site level adoption because the site owns the DNS record of interest.

Reachback, in contrast, does not suffer from these problems. The validation information is maintained by each site, so storage overload is not an issue. This means that Reachback can support almost arbitrary levels of granularity. Further, the finer grain supported by Reachback provides at least a partial solution to the zombie problem. Reachback has better support for incremental adoption. It can be adopted by individual users or groups of users just by establishing their own HTTP server to serve the source keys for that group of users.

The two most important DNS-based systems are DomainKeys and Sender-ID. DomainKeys was invented – and patented – by Yahoo [8,10]. The general operation of DomainKeys is as follows:

1. Extract the *From* header from the email message. Note that DomainKeys, as currently defined, does not use proximate sender information, but it could do so quite easily.
2. Use the domain part of the header to access the corresponding DomainKeys record from some DNS server. Note that DomainKeys

allows for the use of “selectors” to extract a subset of information from the DNS record. The selector can be used to define a limited amount of sub-domain information; the limitation is that it must avoid overloading DNS.

3. Extract the public key for the domain (or sub-domain) from the DNS record.
4. Decrypt the digital signature embedded into the email message (as a special header line) to produce a checksum.
5. Compare the locally computed checksum to the decrypted checksum and if they match, then the email is validated.

Reachback is an explicit alternative to DomainKeys, so it is useful to delimit the differences between the two approaches.

1. Reachback extracts information from an HTTP server specific to the originating source; DomainKeys uses the Domain Name System (DNS) to store validation information.
2. Reachback can utilize a variety of servers to provide Reachback validation information; it can even delegate the validation information to servers outside of its domain.
3. Reachback supports very fine grain validation: down to the level of individual email addresses.
4. Reachback ignores *From* headers and so is relatively immune to in-transit modifications; such modifications to the *From* header can cause DomainKeys to erroneously signal a forgery;
5. Reachback has the potential (Section 12) to utilize validation information other than public/private key pairs, even information private to the Reachback site; DomainKeys is effectively limited to the use of key pairs for validation in order to avoid overloading DNS.

Sender-ID [3,4,14] is a Microsoft patented solution that is also DNS-based. It is derived from earlier work called RMX [17] and SPF (Sender-Policy Framework) [20]. Many of the differences listed above also apply to Sender-ID.

Sender-ID utilizes the remote IP address of the proximate sender of the email. This information is available for any TCP connection and is reliable. This IP address is compared to a list of legal email server machines for the sender site as defined in the site’s DNS record. If the sender is not allowed, or is blacklisted, then validation fails. Aside from the DNS-based problems, Sender-ID also had difficulty with email forwarding and relaying because those mechanisms make it hard to determine the original source of the email.

RMX++ [17] is another DNS-based anti-spam system very much like Sender-ID. It adds one important feature; instead of placing the policy information in DNS directly, it places a URL in DNS and the URL is used to obtain additional DNS records. It still requires DNS, but there is no danger of overloading it. As with Reachback, it uses an HTTP server to store validation information, although the information is not a key, but rather Sender-ID style information. It avoids many of the DNS problems although it still requires site-wide adoption.

PGP Mail has already been described, but it can also be used for blacklisting. A DNS-like key repository network exists for PGP keys. A public key owner registers the key with the repository. Unfortunately, it appears that at the moment there is no verification associated with the registry key. This means that an email receiver has no confidence in the identity of the key owner. In addition, there is no limit to the number of keys that can be registered, so a spammer can in theory provide a different key for every email. This makes blacklisting extremely difficult.

14. STATUS

This system is in the process of being prototyped, so performance and scale data are not yet available. The prototypes are being derived from existing well-known email

components. Open source components are being used where feasible.

The server side will use an SMTP proxy. The receiver side prototype will be done using either a Postfix server combined with a validation proxy or a POP3/IMAP wrapper combined with a validation proxy.

Several metrics must be collected in order to determine the scalability of this approach. One metric will be the cost to reach back to an HTTP server. Another metric will be the required cache size needed to cover some large fraction of received email.

15. SUMMARY

Reachback provides a unique new approach to the spam suppression problem. Unlike competing approaches it does not use DNS. It can use the existing email distribution system, it is robust against in-transit modifications to the standard *From* headers and mail forwarding, and most importantly, it can identify sources down to the granularity of individual users. It has the option to use single secret keys instead of more costly public/private key pairs. It can be incrementally deployed by individual users and can provide incremental value through automated whitelisting.

16. REFERENCES

[1] AHB, The Abusive Hosts Blocking List website. (http://ahbl.org/).

[2] Apache, Spamassassin Project website. (http://spamassassin.apache.org).

[3] Atkinson, R., "Reducing unwanted and unsolicited electronic messages by exchanging electronic message transmission policies and solving and verifying solutions to computational puzzles." US patent application 20040181585, Sept. 16, 2004 (http://uspto.gov/).

[4] Atkinson, R., "Reducing unwanted and unsolicited electronic messages by preventing connection hijacking and domain spoofing." US patent application 20040181571, Sept. 16, 2004 (http://uspto.gov/).

[5] Berners-Lee, T., Fielding, R., and Masinter, L., "Uniform Resource Identifier (URI): Generic Syntax." IETF Request for Comments 3986, Jan. 2005 (http://tools.ietf.org/html/3986).

[6] Boneh, D., "The Difficulties of Tracing Spam Email." FTC Expert Report, Sept. 9, 2004. (http://www.ftc.gov/reports/rewardsys/expertrpt_boneh.pdf).

[7] Cobb, C., "Method and system for filtering electronic messages." US patent number 6,199,102, Mar. 6, 2001 (http://uspto.gov/).

[8] Delany, M., "Method and system for authenticating a message sender using domain keys." US patent number 6,986,049, Jan. 10, 2006 (http://uspto.gov/).

[9] Dmoz, List of blacklist providers. (http://dmoz.org/Computers/Internet/Abuse/spam/Blacklists/).

[10] Hansen, T., Crocker, D., Hallam-Baker, P., "DomainKeys Identified Mail (DKIM) Service Overview." Internet-Draft, June 25, 2006 (http://www.ietf.org/internet-drafts/draft-ietf-dkim-overview-01.txt).

[11] Harris, E., "The Next Step in the Spam Control War: Greylisting." (http://projects.puremagic.com/greylisting/whitepaper.html).

[12] Heiner, J., "Filter-in method for reducing junk e-mail." US patent number 6,112,227, Aug. 29, 2000 (http://uspto.gov/).

[13] Levine, J., "DNS Based Blacklists and Whitelists for E-Mail." Internet Draft draft-irtf asrg-dnsbl-02, Taughannock Networks, Nov. 22, 2005, (http://www.potaroo.net/ietf/idref/draft-irtf-asrg-dnsbl/).

[14] Microsoft, Sender-ID website. (http://www.microsoft.com/mscorp/safety/technologies/senderid/default.mspx).

[15] Naraine, R., "Pump-and-Dump Spam Surge Linked to Russian 'Bot Herders." eWeek.com, November 16, 2006 (http://www.eweek.com/article2/0,1895,2060235,00.asp)

[16] PGP Corporation, PGP website (http://www.pgp.com).

[17] RMX, RMX++/RMX website (http://www.danisch.de/work/security/antispam.html).

[18] SpamCop, SpamCop website (http://www.spamcop.net).

[19] SpamHaus, SpamHaus website (http://www.spamhaus.org/).

[20] SPF, SPF website. (http://www.openspf.org/).