TrustedRelays

Trusted relays, and how header trust works

SpamAssassin will automatically attempt to figure out which Received: headers were inserted by trustworthy mailservers, and which were not. This allows it to:

- optimize DNSBL lookups
- detect when mails never left a trusted network path
- know when a Received header can be trusted for whitelisting purposes
- produce synthetic 'pseudo-headers', allowing rules to match against the message's network traversal portably

This page details the concept of 'trust' internally to SpamAssassin, and how it appears in the output. See also TrustPath for details on how to influence this by setting the 'trusted_networks' parameter, and DynablockIssues for authenticated mail submission issues.

Note that the trust path information is used by both network and non-net rules, so even if you're not running with network rules enabled (the "-L" switch), it's worth configuring this.

An example

Here's an example email, with sets of headers for analysis:

```
Received: from internal.example.com [127.0.0.1] by localhost
    for someone@example.com; Fri, 07 Dec 2001 11:07:40 +1100 (EST)
Received: from dmz.example.com [150.51.53.1] by internal.example.com
    for someone@example.com; Fri, 07 Dec 2001 11:07:35 +1100 (EST)
Received: from friend.example.com [212.17.35.14] by dmz.example.com
    for someone@example.com; Fri, 07 Dec 2001 11:07:35 +1100 (EST)
Received: from notrust.example.com [193.120.149.226] by friend.example.com
    for someone@example.com; Fri, 07 Dec 2001 11:07:30 +1100 (EST)
Received: from loser.example.org [61.119.13.18] by notrust.example.com
    for someone@example.com; Fri, 07 Dec 2001 11:07:25 +1100 (EST)
Received: from chaos.example.net [210.73.88.134] by loser.example.org
    for someone@example.com; Fri, 07 Dec 2001 11:07:20 +1100 (EST)
Received: from evil.example.net [144.137.3.98] by chaos.example.net
    for someone@example.com; Fri, 07 Dec 2001 11:07:15 +1100 (EST)
From: "DNSBL Testing" <spammer@example.com>
Subject: no subject needed
Date: Fri, 7 Dec 2001 07:01:03
Message-Id: <20011206235802.4FD6F1143D6@mail.netnoteinc.com>

hello
```

Assuming SpamAssassin is running on 'internal.example.com', and the TrustPath on that machine is set up to trust the DMZ machine 'dmz.example.com' at 150.51.53.1 (and also consider that internal using `internal_networks`), and a trustworthy external machine 'friend.example.com' at 212.17.35.14, this means that the message passed through the following relays:

- **Untrusted** source at evil.example.net [144.137.3.98]
- **Untrusted** relay at chaos.example.net [210.73.88.134]
- **Untrusted** relay at loser.example.org [61.119.13.18]
- **Untrusted** relay at notrust.example.com [193.120.149.226]
- **Trusted** relay at friend.example.com [212.17.35.14]
- **Trusted** and **internal** relay at dmz.example.com [150.51.53.1]
- **Trusted** and **internal** localhost handover at internal.example.com [127.0.0.1]

A side note: the header lines for 'evil', 'chaos', and 'loser' could all be faked, for all we know, since who knows if an untrusted host is running legitimate MTA software, or is under the control of a spammer? Therefore, it's unwise to trust things you find in untrusted headers. (One exception to this is discussed at the end of this article.)

Here's what SpamAssassin's trust path algorithm makes of this (pasted from "spamassassin -D -t" output): TrustedRelaysDebugExample.

The 'X-Spam-Relays' Pseudo-headers

The last few lines of that debug output are most noteworthy – especially since rules can match against these **metadata pseudo-headers**. Here they are:

the 'X-Spam-Relays-Trusted' pseudoheader:

```
[ ip=127.0.0.1 rdns= helo=internal.example.com by=localhost ident= envfrom= intl=1 id= auth= ]
[ ip=150.51.53.1 rdns= helo=dmz.example.com by=internal.example.com ident= envfrom= intl=1 id= auth= ]
[ ip=212.17.35.14 rdns= helo=friend.example.com by=dmz.example.com ident= envfrom= intl=0 id=auth= ]
```

the 'X-Spam-Relays-Untrusted' pseudoheader:

```
[ ip=193.120.149.226 rdns= helo=notrust.example.com by=friend.example.com ident= envfrom= intl=0 id= auth= ]
[ ip=61.119.13.18 rdns= helo=loser.example.org by=notrust.example.com ident= envfrom= intl=0 id= auth= ]
[ ip=210.73.88.134 rdns= helo=chaos.example.net by=loser.example.org ident= envfrom= intl=0 id= auth= ]
[ ip=144.137.3.98 rdns= helo=evil.example.net by=chaos.example.net ident= envfrom= intl=0 id= auth= ]
```

There are also two more – 'X-Spam-Relays-Internal' and 'X-Spam-Relays-External'.

They are divided into trusted/untrusted and internal/external pairs, depending on the setting of 'trusted_networks' and 'internal_networks'.

You can see they list the contents of the Received header in a machine-readable, and standardised, format, so that rules can be insulated from the vagaries of the Received header, which has a tendency to look radically different between MTAs. (Some MTAs even reverse the order of the items, but look otherwise identical!)

In the samples above, they include newlines; however, in the real pseudo-headers produced by SpamAssassin, each [...] block is simply space-separated.

Some sample rules that use this data can be seen in the standard SpamAssassin rules file, '20_fake_helo_tests.cf'. Here is an example:

```
header RULENAME  X-Spam-Relays-Untrusted =~ /^[^\]]+ helo=foo /i
```

DNSBL lookups and the most recent untrusted host

DNSBL rules support '-firsttrusted' and '-untrusted' as special-case keywords to control IP address selection. These keywords do not refer to the trust status of the lines themselves! They refer to the trust status of the data that will be looked up in the DNSBL.

This hinges on a key border case. The most recent 'untrusted' header line is in an interesting grey area – the **host** it discusses is an untrusted host, but the **data** recorded about that host is, in itself, trustworthy.

Above, for example, 193.120.149.226 is listed as an untrusted host and is therefore listed in the 'X-Spam-Relays-Untrusted' pseudoheader. However, its IP address was *recorded* by a trusted host, so the IP address data *is* trustworthy.

This is the most commonly tested item in the string, since it's the most likely host to be a spam zombie or spammer MTA.

Testing the "most recent untrusted" host in a header rule is done as follows:

```
  header RULENAME  X-Spam-Relays-Untrusted =~ /^[^\]]+ helo=foo /i
```

Note the `^[^]]+` part of the pattern; that skips anything but "]" characters, ensuring that the match will only happen within the first [...] block of the pseudo-header string.

Checking that IP address in a DNSBL lookup using check_rbl() is performed by appending the string '-firsttrusted' to the set name:

```
  header RULENAME  eval:check_rbl_txt('bsp-firsttrusted', 'sa-trusted.bondedsender.org.', '(?i:bonded)')
```

Using Other Untrusted Hosts

The 'most recent untrusted' host is the only 'grey area', however. All the other hosts listed in the 'X-Spam-Relays-Untrusted' pseudoheader were both untrusted themselves, and their details were not recorded by a trusted host; both the lines themselves and the IP addresses are not trustworthy, since they could have been generated by a spamware application creating fake header data. It's especially important not to trust that data for rules that could give negative points, since spammers can, and will, attempt to fake their way around your whitelisting rules.

Also worth noting: it's common for the "trusted" networks to extend further than the "internal" networks. If you are writing rules to match the host which delivered a mail into the SMTP MX server, you should use "external" instead of "untrusted", since it's common for "good" third-party senders to be put into the "trusted" list. (This is especially important for rules that match features of *dynamic host* senders, such as rDNS patterns etc.)