**MIMECAST, GSUITE – INBOUND DKIM / SPF & DMARC Rejects**

**SETUP:**

Mimecast email filtering services in front of GSuite

**ISSUE:**

Some incoming emails, from reputable providers (questar.org) are being hard bounced by GSuite after being processed by the inbound Mimecast filters

**ERROR MESSAGE FROM MIMECAST:**

*5.7.26 Unauthenticated email from questar.org is not accepted due to  
550-5.7.26 domain's DMARC policy. Please contact the administrator of  
550-5.7.26 questar.org domain if this was a legitimate mail. To learn about the  
550-5.7.26 DMARC initiative, go to  
550 5.7.26*[*https://support.google.com/mail/?p=DmarcRejection*](https://support.google.com/mail/?p=DmarcRejection)*k10-20020a05620a414a00b00787d387de35si249645qko.47 – gsmtp*

**WHAT IS ACTUALLY HAPPENING?**

Email arrives at Mimecast. It is “exploded”, inspected and then repacked for onward delivery to GSuite / GMail.

**IF** the sender has DKIM signed the email, then this “explode, inspect and repack” breaks the DKIM signature.

Looking at email headers (in this case the sender being an outlook.com address) you see this sort of thing:

*dkim=neutral (****body hash did not verify****) header.i=@*[outlook.com](http://outlook.com/) *header.s=selector1 header.b=DEOq8NQm*

When the email is then handed to GSuite, this broken DKIM signature / body hash will cause issues in GSuite.

GSuite, when faced with a failing DKIM signature, looks up the DMARC settings for the **sender** domain.

* If there is no DMARC – we presume it lets it through
* If the DMARC is set to “none” – GSuite lets it through
* If the DMARC is set to “quarantine” – We don’t know what GSuite does. Probably rejects it (unless you’ve configured quarantine functionality)
* If the DMARC is set to “reject” – then GSuite hard bounces the email and generates the 5.7.1 error that we see in the Mimecast console (see above for an example).

This “reject” option is set by organizations like questar.org as it’s a sensible anti-tamper/anti-spoofing protection for email communications.

**HOW TO FIX:**

1- Don’t get Mimecast to “explode, inspect and repack”, so the DKIM signature isn’t broken. Kinda pointless having Mimecast at all in that case. Mimecast’s usually excellent L1 support seems to think that exempting these domains from the “inbound DNS checks” is the way to fix this. We disagree, having to explicitly whitelist domains is not where we want to be. Plus, it’s a hard bounce from GSuite, so you’ve got to phone and ask the sender to retry / resend every single time something gets caught.

2- Get GSuite to ignore the “reject” instruction in the DMARC policy. We don’t see that ignoring an explicit “reject” instruction is a good place to be.

3-  Get GSuite to not worry about the broken DKIM and so not bother looking up the DMARC policy of the sender.

So – option 3 it is then.

**HOW?**

This is the part that is absent from the Mimecast documentation that we were sent for Mimecast implementation with GSuite. It's vaguely mentioned in the "lock down your GSuite" section - but it's not at all clear that this is what is needed to for DMARC etc

**IN THE GSUITE ADMIN PANEL:**

Apps -> G Suite -> Settings for Gmail -> Advanced settings

You’re looking for the section that says “*inbound gateway*“.

This is the functionality that is explicitly designed for this situation.

You need to add the Mimecast IP ranges for our region. YMMV.

*Inbound gateway*  
*Locally applied*  
*Mimecast Gateway IP(s): 195.130.217.0/24, 91.220.42.0/24, 185.58.84.0/22, 146.101.78.0/24,* 207.82.80.0/24 *Require Inbound Gateway IP: NoRequire Secure (TLS) Connections: NoDisable Gmail Spam Filtering: No*

Once set up and applied – this will (as confirmed by GSuite support) stop GSuite panicking about the broken DKIM.

This set up essentially exempts emails that arrive via Mimecast from the DKIM checks. (and presumably SPF as well)

**RESULT:**

Senders with strict “reject” DMARC policies can now successfully deliver inbound to GSuite, even though Mimecast breaks their DKIM signed emails. It’ll also help stop GSuite making a poor decision around SPF record checking.