[messaging] Modern anti-spam and E2E crypto

Mike Hearn mike at plan99.net
Fri Sep 5 08:07:30 PDT 2014

- Previous message: [messaging] Opportunistic encryption and authentication methods
- Next message: [messaging] Modern anti-spam and E2E crypto
- Messages sorted by: [date] [thread] [subject] [author]

```
Hey,

Trevor asked me to write up some thoughts on how spam filtering and fully
end to end crypto would interact, so it's all available in one message
instead of scattered over other threads. Specifically he asked for brain
dumps on:

   - how does antispam currently work at large email providers
   - how would widespread E2E crypto affect this
   - what are the options for moving things to the client (and pros, cons)
   - is this feasible for email?
   - How do things change when moving from email to other sorts of
async messaging
   (e.g. text messaging) or new protocols - i.e. are there unique aspects
   of existing email protocols, or are these general problems?

Brief note about my background, to establish credentials:  I worked at
Google for about 7.5 years. For about 4.5 of those I worked on the Gmail
abuse team, which is very tightly linked with the spam team (they use the
same software, share the same on-call rotations etc).

Starting around mid-2010 we had put sufficient pressure on spammers that
they were unable to make money using their older techniques, and some of
them switched to performing industrial-scale hacking of accounts using
compromised passwords (and then sending spam to the account's contacts), so
I became tech lead of a new anti-hijacking team. We spent about 2.5 years
beating the hijackers. In early 2013 we declared victory
<http://googleblog.blogspot.ch/2013/02/an-update-on-our-war-against-account.html>
and
a few months later, Edward Snowden revealed that the NSA/GCHQ was tapping
the security system we had designed
<http://www.theguardian.com/technology/2013/nov/06/google-nsa-gchq-spying-judicial-process>
.

Since then things seem to be pretty quiet. It's not implausible to say that
from Gmail's perspective the spam war has been won .... for now, at least.

In case you prefer videos to reading a few years ago I gave a talk at the
RIPE64 conference in Ljubljana:

   https://ripe64.ripe.net/archives/video/25/

In January I left Google to focus on Bitcoin full time. My current project
is a p2p crowdfunding app I want to use as a way to fund development of
decentralised infrastructure.

OK, here we go.
```

A brief history of the spam war

In the beginning ... there was the regex. Gmail does support regex filtering but only as a last resort. It's easy to make mistakes, like the time we accidentally blackholed email for an unfortunate Italian woman named "Oli*via Gra*dina". Plus this technique does not internationalise, and randomising text to miss the blacklists is easy.

The email community began sharing abusive IPs. Spamhaus was born. This approach worked better because it involved burning something that the spammer had to pay money to obtain. But it caused huge fights because the blacklist operators became judge, jury and executioner over people's mail streams. What spam actually is turned out to be a contentious issue. Many bulk mailers didn't think they were spamming, but in the absence of a clear definition sometimes blacklisters disagreed.

Botnets appeared as a way to get around RBLs, and in response spam fighters mapped out the internet to create a "policy block list" - ranges of IPs that were assigned to residential connections and thus should not be sending any email at all. Botnets generate enormous amounts of spam by volume, but it's also the easiest spam to filter. Very little of my time on the Gmail spam/abuse team was spent thinking about botnets.

Webmail services like Gmail came on the scene. The very first release of Gmail simply used spamassassin on the backend, but this was quickly deemed not good enough and a custom filter was built. The architect of the Gmail filter wrote a paper in 2006 which you can find here:

   http://ceas.cc/2006/19.pdf

I'll summarise it. The primary technique the new filter used was attempting to heuristically guess the sending domain for email (domains being harder to obtain and more stable than IPs), and then calculating *reputations* over them. A reputation is a score between 0-100 where 100 is perfectly good and 0 means always spam. For example if a sender had a reputation of 70 that means about 30% of the time we think their mail is spam and the rest of the time it's legit. Reputations are moving averages that are calculated based on a careful blend of manual feedbacks from the Report Spam/Not Spam buttons and "auto feedbacks" generated by the spam filter itself. Obviously, manual feedbacks have a lot more weight in the system and that allows the filter to self correct.

This approach has another advantage - it eliminates all the political fighting. The new definition of spam is "whatever our users say spam is", a definition that cannot be argued with and is simultaneously crisp enough to implement, yet vague enough to adapt to whatever spammers come up with.

It's worth noting a few things here:

- Reputation systems require the ability to read *all* email. It's not good enough to be able to see only spam, because otherwise the reputations have no way to self correct. The flow of "not spam" reports is just as important as the flow of spam reports. Most not spam reports are generated implicitly of course, by the act of not marking the message at all.

- You need to calculate reputations *fast*. If you receive mail with unknown reputations, you have no choice but to let it pass as otherwise you can't figure out if it's spam or not. That in turn incentivises spammers to try and outrun the learning system. The first version of the reputation system used MapReduce and calculated reputations in batch, so convergence took hours. Eventually it had to be replaced with an online system that recalculates scores on the fly. This system is a tremendously impressive piece of engineering - it's basically a global, real time peer to peer learning system. There are no masters. The filter is distributed throughout the world and can tolerate the loss of multiple datacenters.

I don't want to think about how you'd build one of these outside a highly controlled environment, it was enough of a headache even in the proprietary/centralised setting ....

- Reputations propagate between each other. If we know a link is bad and it appears in mail from an IP with unknown reputation, then that IP gets a bad reputation too and vice versa. It turns out that this is important - as the number of things upon which reputations are calculated goes up, it becomes harder and harder for spammers to rotate all of them simultaneously. Especially this is true if using a botnet where precise control over the sending machines is hard. If a spammer fails to randomize even one tiny aspect of their mail at the same time as the others, all their links and IPs get automatically burned and they lose money.

- Reputation contains an inherent problem. You need lots of users, which implies accounts must be free. If accounts are free then spammers can sign up for accounts and mark their own email as not spam, effectively doing a sybil attack on the system. This is not a theoretical problem.

The reputation system was generalised to calculate reputations over *features* of messages beyond just sending domain. A message feature can be, for example, a list of the domains found in clickable hyperlinks. Links would turn out to be a critical battleground that would be extensively fought over in the years ahead. The reason is obvious: spammers want to sell something. Therefore they must get users to their shop. No matter how they phrase their offer, the URL to the destination must work. The fight went like this:

1. They start with clear clickable links in HTML emails. Filters start blocking any email with those links.

2. They start obfuscating the links, and requesting users put the link back together. But this works poorly because many users either can't or won't figure it out, so profits fall.

3. They start buying and creating randomised domains in bulk. TLDs like .com are expensive but others are cheap or free and the reputations of the entire TLDs went into freefall (like .cc)

4. Spammers run out of abusable TLDs as registrars begin to crack down. They begin performing *reputation hijacking*, e.g. by creating blogs on sites which allow you to register *.blogspot.com, *.livejournal.com and so on. URL shorteners become a spammers best friend. Literally every URL shortener immediately becomes a war zone as the operators and spammers fight to defend and attack the URL domain reputations.

5. Spammers also start hacking websites but this doesn't work that well, because many websites don't often appear in legitimate mail often so they don't have strong reputations. Great source of passwords though.

6. Big content hosting sites like Google begin connecting their spam filters to their hosting engines so once the reputation of a user-generated URL falls it's automatically terminated. The first iterations of this are too slow. One of my projects at Google was to build a real-time system to do this automatic content takedown.

Obtaining fresh sending IP addresses was a problem for them too of course. The best fix was to use webmail services as anonymizing proxies. Gmail was hit especially hard by this because early on Paul Buchheit (the creator) decided not to include the client IP address in email headers. This was either a win for user privacy or a blatant violation of the RFCs, depending on who you asked. It also turned Gmail into the worlds biggest anonymous remailer - a real asset for spammers that let them sail right past most filters which couldn't block messages from a sender as large as Google.

Between about 2006 (open signups) and 2010 a lot of the anti-spam work involved building a spam filter for account signups. We did a pretty good job, even though I say so myself. You can see the prices of different kinds of "free" webmail accounts at http://buyaccs.com (a Russian account shop). Note that hotmail/outlook.com accounts cost $10 per thousand and gmails cost an order of magnitude more. When we started gmails were about $25 per 1000 so we were able to quadruple the price. Going higher than that is hard because all big websites use phone verification to handle false positives and at these price levels it becomes profitable to just buy lots of SIM cards and burn phone numbers.

There's a significant amount of magic involved in preventing bulk signups. As an example, I created a system that randomly generates encrypted JavaScripts that are designed to resist reverse engineering attempts. These programs know how to detect automated signup scripts and entirely wiped them out
<http://webcache.googleusercontent.com/search?q=cache:v6Iza2JzJCwJ:www.hackforums.net/archive/index.php/thread-2198360.html+&cd=8&hl=en&ct=clnk&gl=ch>
.

How would widespread E2E crypto affect all this

You can see several themes in the above story:

- Large volumes of data is really important, of both legit and spam messages.
- Extremely high speed is important. A lot of spam fights boil down to a game of who is faster. If your reputations converge in 3 minutes then you're going to be outrun.
- Being able to police your user base is important. You can't establish reputations if you can't trust your user reports and that means creating a theoretically impossible situation: accounts that are free yet also cost money (if you need lots of them)

The first problem we have in the E2E context is that reputation databases require input from *all* mail. We can imagine an email client that knows how to decrypt a message, performs feature extraction and then uploads a "good mail" or "bad mail" report to some <handwave> central facility. But then that central facility is going to learn not only who you are talking with but also what links are in the mail. That's probably quite valuable information to have. As you add features this problem gets worse.

The second problem we have is that if the central reputation aggregator can't read your mails, it doesn't know if you did feature extraction honestly. This is not a problem in the unencrypted context because the spam filter extracts features itself. Whilst spammers can try to game the system, they still have to actually send their spams to themselves for real, and this imposes a cost. In a world where spam filters cannot read the message, spammers can just submit entirely fictional "good mail" reports. Worse, competitors could interfere with each others mail streams by submitting false reports. We see this sort of thing with AdWords.

The third problem is that spam filters rely quite heavily on security through obscurity, because it works well. Though some features are well known (sending IP, links) there are many others, and those are secret. If calculation was pushed to the client then spammers could see exactly what they had to randomise and the cross-propagation of reputations wouldn't work as well.

It might be possible to resolve the above two problems using trusted computing. With TC you can run encrypted software on private data and the hardware will "prove" what it ran to a remote server. But security through obscurity and end to end crypto are hard to mix - if you run your email content through a black box, that black box could potentially steal the contents. You have to trust the entity calculating the secret sauce with your message, and then you could just use Gmail in the regular way as today.

The fourth problem we have is that anonymous usage and spam filters don't really mix. Ultimately there's no replacement for cutting spam off at the source. Account termination is a fundamental spam fighting tool. All major webmail and social services force users to perform phone verification if they trip an abuse filter. This sends a random code via SMS or voice call to a phone number and verifies the user can receive it. It works because phone numbers are a resource that have a cost associated with them, yet ~all users have one. But in many countries it's illegal to have anonymous mobile numbers and operators are forced to do ID verification before handing out a SIM card. The fact that you can be "name checked" at any moment with plausible deniability means that whilst you don't have to provide any personal data to get a webmail account, a government could force you to reveal your location and/or identity at any time. They don't even have to do anything special; if they can phish your password they can forcibly trip the abuse filter, wait for the user to pass phone verification, then get a warrant for the users account metadata knowing that it now contains what they need (I never saw any evidence of this, but it's theoretically possible).

The final problem we have is that spam filtering is resource intensive CPU and disk wise. Many, many users now access their email *exclusively* via a smartphone. Smartphones do not have many resources and the more work you do, the worse the battery life. Simply waking up the radio to download a message uses battery. Attempting to do even obsolete 1990's style spam filtering of all mail received with a phone would probably be a non starter unless there's some fundamental breakthrough in battery technology.

In conclusion, I don't see a return to pure client side filtering being feasible.

How do things change when moving from email to other sorts of async messaging ?

Well. SMS spam is a thing. It doesn't happen much because phone companies act as spam filters. Also, because governments tend to get involved with the punishment of SMS spammers, in order to discourage copycat offenders and send a message (pun totally intended). Email spam blew up way before governments could react to it, so it's interesting to see the different paths these systems have taken.

Systems like WhatsApp don't seem to suffer spam, but I presume that's just an indication that their spam/abuse team is doing a good job. They are in the easiest position. When you have central control everything becomes a million times easier because you can change anything at any time. You can terminate accounts and control signups. If you don't have central control, you have to rely exclusively on inbound filtering and have to just suck it up when spammers try to find ways around your defences. Plus you often lose control over the clients.

General thoughts and conclusions

When you look at what it's taken to win the spam war with cleartext, it's been a pretty incredible effort stretched over many years. "War" is a good analogy: there were two opposing sides and many interesting battles, skirmishes tactics and weapons. I could tell stories all day but this email is already way too long.

Trying to refight that in the encrypted context would be like trying to fight a regular war blindfolded and handcuffed. You'd be dead within minutes.

So I think we need totally new approaches. The first idea people have is to make sending email cost money, but that sucks for several reasons; most obviously - free global communication is IMHO one of humanities greatest achievements, right up there with putting a man on the moon. Someone from rural China can send me a message within seconds, for free, and I can reply, for free! Think about that for a second.

The other reason it sucks is that it confuses bulk mail with spam. This is a very common confusion. Lots of companies send vast amounts of mail that users want to receive. Think Facebook, for example. If every mail cost money, some legit and useful businesses wouldn't work, let alone things like mailing lists.

A possibly better approach is to use money to create deposits. There is a protocol that allows bitcoins to be sacrificed to miners fees, letting you prove that you threw money away by signing challenges with the keys that did so. This would allow very precise establishment of an anonymous yet costly credential that can then send as much mail as it wants, and have reputations calculated over it. Spam/not spam reports that *only* contain proof of sending could then be scatter/gathered and used to calculate a reputation, or if there is none, then such mails could be throttled until a few volunteers have peeked inside. Another approach would be to allow cross-signing - an entity with good reputation can temporarily countersign mail to give it a reputational boost and trigger cross-propagation of reputations. That entity could employ whatever techniques they liked to verify the senders legitimacy.

It's for these reasons that I'm interested in the overlap between Bitcoin and E2E messaging. It seems to me they are fundamentally linked.

Final thought. I'm somewhat notorious in the Bitcoin community for making radical suggestions, like maybe there exists a tradeoff between privacy and abuse. Lots of people in the crypto community passionately hate this idea and (unfortunately) anyone who makes it. I guess you can see based on the above stories why I think this way though. It's not clear to me that chasing perfect privacy whilst ignoring abuse is the right path for any system that wishes to achieve mainstream success.
-------------- next part --------------
An HTML attachment was scrubbed...
URL: <http://moderncrypto.org/mail-archive/messaging/attachments/20140905/e09e4700/attachment.html>

- Previous message: [messaging] Opportunistic encryption and authentication methods
- Next message: [messaging] Modern anti-spam and E2E crypto
- Messages sorted by: [date] [thread] [subject] [author]

More information about the Messaging mailing list