A Few Thoughts on Cryptographic Engineering

Some random thoughts about crypto. Notes from a course I teach. Pictures of my dachshunds.

Matthew Green in attacks, messaging January 10, 2018January 10, 2018 1,984 Words

Attack of the Week: Group Messaging in WhatsApp and Signal

If you've read this blog before, you know that secure messaging is one of my favorite topics. However, recently I've been a bit disappointed. My sadness comes from the fact that lately these systems have been getting *too damned good*. That is, I was starting to believe that most of the interesting problems had finally been solved.

If nothing else, today's post helped disabuse me of that notion.

This result comes from a new paper (https://eprint.iacr.org/2017/713.pdf) by Rösler, Mainka and Schwenk from Ruhr-Universität Bochum (affectionately known as "RUB"). The RUB paper paper takes a close look at the problem of *group messaging*, and finds that while messengers may be doing fine with normal (pairwise) messaging, group messaging is still kind of a hack.

If all you want is the TL;DR, here's the headline finding: due to flaws in both Signal and WhatsApp (which I single out because I use them), it's *theoretically* possible for strangers to add themselves to an encrypted group chat. However, the caveat is that these attacks are extremely difficult to pull off in practice, so nobody needs to panic. But both issues are very avoidable, and tend to undermine the logic of having an end-to-end encryption protocol in the first place. (Wired also has a good article (https://www.wired.com/story/whatsapp-security-flaws-encryption-group-chats/).)

First, some background.

How do end-to-end encryption and group chats work?

In recent years we've seen (https://en.wikipedia.org/wiki/Sony_Pictures_hack) plenty of evidence (https://en.wikipedia.org/wiki/2016_Democratic_National_Committee_email_leak) that centralized messaging servers aren't a very good place to store confidential information. The good news is: we're not stuck with them. One of the most promising advances in the area of secure communications has been the recent (https://signal.org/blog/whatsapp/) widespread (https://core.telegram.org/api/end-to-end) deployment of *end-to-end (e2e) encrypted messaging protocols.*

At a high level, e2e messaging protocols are simple: rather than sending plaintext to a server — where it can be stolen or read — the individual endpoints (typically smartphones) *encrypt* all of the data using keys that the server doesn't possess. The server has a much more limited role, moving and storing only meaningless ciphertext. With plenty of caveats (https://www.lawfareblog.com/apple-calea-and-law-enforcement), this means a corrupt server shouldn't be able to eavesdrop on the communications.

In pairwise communications (*i.e.,* Alice communicates with only Bob) this encryption is conducted using a mix of public-key and symmetric key algorithms. One of the most popular mechanisms is the Signal protocol (https://eprint.iacr.org/2016/1013.pdf), which is used by Signal (https://signal.org/) and WhatsApp (https://www.whatsapp.com/) (notable for having 1.3 *billion* users!) I won't discuss the details of the Signal protocol here, except to say that it's complicated, but it works pretty well.

A fly in the ointment is that the standard Signal protocol doesn't work quite as well for *group messaging,* primarily because it's not optimized for broadcasting messages to many users.

To handle that popular case, both WhatsApp and Signal use a small hack. It works like this: each group member generates a single "group key" that this member will use to encrypt all of her messages to everyone else in the group. When a new member joins, everyone who is already in the group needs to send a copy of their *group key* to

the new member (using the normal Signal pairwise encryption protocol). This greatly simplifies the operation of group chats, while ensuring that they're still end-to-end encrypted.

How do members know when to add a new user to their chat?

Here is where things get problematic.

From a UX perspective, the idea is that only one person actually initiates the adding of a new group member. This person is called the "administrator". This administrator is the only human being who should actually *do* anything — yet, her one click must cause some automated action on the part of every other group members' devices. That is, in response to the administrator's trigger, *all devices in the group chat* must send their keys to this new group member.

(In Signal, every group member is an administrator. In WhatsApp it's just a subset of the members.)

The trigger is implemented using a special kind of message called (unimaginatively) a "group management message". When I, as an administrator, add Tom to a group, my phone sends a group management message to all the existing group members. This instructs them to send their keys to Tom — and to *notify* the members visually so that they know Tom is now part of the group. Obviously this should only happen if I *really did* add Tom, and not if some outsider (like that sneaky bastard Tom himself!) tries to add Tom.

And this is where things get problematic.

Notification messages in WhatsApp.

Ok, what's the problem?

According to the RUB paper (https://eprint.iacr.org/2017/713.pdf), both Signal and WhatsApp fail to properly authenticate group management messages.

The upshot is that, at least in theory, this makes it possible for an unauthorized person — not a group administrator, possibly not *even a member of the group* — to add someone to your group chat.

The issues here are slightly different between Signal and WhatsApp. To paraphrase Tolstoy, every working implementation is alike, but every broken one is broken in its own way. And WhatsApp's implementation is somewhat worse than Signal. Here I'll break them down.

Signal. Signal takes a pragmatic (and reasonable) approach to group management. In Signal, every group member is considered an administrator — which means that any member can add a new member. Thus if I'm a member of a group, I can add a new member by sending a group management message to every other member. These messages are sent encrypted via the normal (pairwise) Signal protocol.

The group management message contains the "group ID" (a long, unpredictable number), along with the identity of the person I'm adding. Because messages are sent using the Signal (pairwise) protocol, they should be *implicitly* authenticated as coming from me — because authenticity is a property that the pairwise Signal protocol already offers. So far, this all sounds pretty good.

The problem that the RUB researchers discovered through testing, is that while the Signal protocol *does* authenticate that the group management comes from me, it doesn't actually check that *I am a member of the group* — and thus authorized to add the new user!

In short, if this finding is correct, it turns out that *any random Signal user in the world* can you send a message of the form "Add Mallory to the Group 8374294372934722942947", and (if you happen to belong to that group) your app will go ahead and try to do it.

The good news is that in Signal the attack is very difficult to execute. The reason is that in order to add someone to your group, *I need to know the group ID*. Since the group ID is a random 128-bit number (and is never revealed to non-group-members or even the server**) that pretty much blocks the attack. The main exception to this is *former* group members, who already know the group ID — and can now add themselves back to the group with impunity.

(And for the record, while the group ID may block the attack, it really seems like a lucky break — like falling out of a building and landing on a street awning. There's no reason the app should process group management messages from random strangers.)

So that's the good news. The bad news is that WhatsApp is a bit worse.

WhatsApp. WhatsApp uses a slightly different approach for its group chat. Unlike Signal, the WhatsApp server plays a significant role in group management, which means that it determines who is an administrator and thus authorized to send group management messages.

Additionally, group management messages are not end-to-end encrypted or signed. They're sent to *and from the WhatsApp server* using transport encryption, but not the actual Signal protocol.

When an administrator wishes to add a member to a group, it sends a message to the server identifying the group and the member to add. The server then checks that the user is authorized to administer that group, and (if so), it sends a message to every member of the group indicating that they should add that user.

The flaw here is obvious: since the group management messages are not signed by the administrator, a malicious WhatsApp server *can add any user it wants into the group.* This means the privacy of your end-to-end encrypted group chat is only guaranteed if you actually trust the WhatsApp server.

This undermines the entire purpose of end-to-end encryption.

But this is silly. Don't we trust the WhatsApp server? And what about visual notifications?

One perfectly reasonable response is that *exploiting this vulnerability requires a compromise of the WhatsApp server* (or legal compulsion, perhaps). This seems fairly unlikely.

And yet, the entire point of end-to-end encryption is to remove the server from the trusted computing base. We haven't *entirely* achieved this yet, thanks to things like key servers. But we are making progress. This bug is a step back, and it's one a sophisticated attacker potentially *could* exploit.

A second obvious objection to these issues is that adding a new group member results in a *visual* notification to each group member. However, it's not entirely clear that these messages are very effective. In general they're relatively easy to miss. So these are meaningful bugs, and things that should be fixed.

How do you fix this?

The great thing about these bugs is that they're both eminently fixable.

The RUB paper (https://eprint.iacr.org/2017/713.pdf) points out some obvious countermeasures. In Signal, just make sure that the group management messages come from a legitimate member of the group. In WhatsApp, make sure that the group management messages are *signed* by an administrator.*

Obviously fixes like this are a bit complex to roll out, but none of these should be killers.

Is there anything else in the paper?

Oh yes, there's quite a bit more (https://eprint.iacr.org/2017/713.pdf). But none of it is quite as dramatic. For one thing, it's possible for attackers to block message acknowledgements in group chats, which means that different group members could potentially see very different versions of the chat. There are also several cases where forward secrecy can be interrupted. There's also some nice analysis of Threema, (https://threema.ch/en) if you're interested.

I need a lesson. What's the moral of this story?

The biggest lesson is that protocol specifications are never enough. Both WhatsApp (https://www.whatsapp.com/security/WhatsApp-Security-Whitepaper.pdf) and Signal (to an extent) have detailed protocol specifications that talk quite a bit about the cryptography used in their systems. And yet the issues reported in the RUB paper not obvious from reading these summaries. I certainly didn't know about them.

In practice, these problems were only found through testing.

So the main lesson here is: *test, test, test*. This is a strong argument in favor of open-source applications and frameworks that can interact with private-garden services like Signal and WhatsApp. It lets us see what the systems are getting right and getting wrong.

The second lesson — and a very old one — is that cryptography is only half the battle. There's no point in building the most secure encryption protocol in the world if someone can simply instruct your client *to send your keys to Mallory.* The greatest lesson of all time is that real cryptosystems are always broken this way — and almost never through the fancy cryptographic attacks we love to write about.

Notes:

Mallory.

* The challenge here is that since WhatsApp itself determines *who* the administrators are, this isn't *quite* so simple. But at very least you can ensure that someone in the group was responsible for the addition.

** According to the paper, the Signal group IDs are always sent encrypted between group members and are never revealed to the Signal server. Indeed, group chat messages look exactly like pairwise chats, as far as the server is concerned. This means only current or former group members should know the group ID.

3 thoughts on "Attack of the Week: Group Messaging in WhatsApp and Signal"

HacKan says:
January 10, 2018 at 3:43 pm
Nice post! I've to point out a mistake, nevertheless: "The main exception to this is former group members, who already know the group ID — and can now add themselves back to the group with impunity.": in Signal, you can't remove a member of a group, you need to create a whole new group! I've suffer that lack of possibility, creating groups over and over and over…
What you could do is leave the group and then, by this attack, join back in. Which isn't a great deal anyway.

Cheers!

Reply
Sebastian says:
January 10, 2018 at 4:05 pm
Great post. I yhink I found a minor typo:
"This means the privacy of your end-to-end encrypted group chat is only guaranteed if you actually the WhatsApp server."

Reply
maqp says:
January 11, 2018 at 9:07 am
Thanks for the informative post. What are your thoughts on at what point a protocol stops being some specific protocol when it's being changed? E.g. AES stops being AES if you add one round to Rijndael algorithm. It's not weaker, but it's no longer AES. Considering the major differences, should we talk about WhatsApp's protocol as it's own double-ratchet protocol or as a customized Signal-protocol, instead of saying WhatsApp uses Signal protocol?

Reply

Menu