Hardware Tokens & Digital Signatures

We'd like to improve the security of our software release process by digitally signing files with easy to implement, secure, low cost, modern cryptography using a hardware security token.

Requirements

We would like any solution to satisfy the following requirements as much as possible, in no particular order:

1. Using modern cryptography, i.e. EdDSA / Curve25519, no RSA or ECDSA;
2. No complicated encoding/serialization with for example X.509, PKCS#*, ASN.1, BER/DER, CBOR, …;
3. Implementations SHOULD be implementable on/with open hardware;
4. Cheap, i.e. affordable by any developer, out of pocket if necessary;
5. Easy to write/audit code to *verify* the signatures;
6. Considering existing solutions and possibly improving (on) them;
7. Work with existing projects and gather (additional) requirements;
8. Security and simplicity are more important than features;
9. Ability to sign/verify big files (possibly using pre-hashing);
10. Ideally usable for various applications:
    - document signing;
    - as a replacement for PGP [8] in Linux package distributions [4];
    - software release signing
    - …

We are thinking about a USB/NFC hardware key, as that makes it easily portable, but solutions using e.g. *Touch ID* or other mechanisms integrated in (mobile) operating systems would also be interesting to investigate.

Existing (Partial) Solutions

We are aware of the following potential solutions that match the above requirements to some extent:

- YubiKey OpenPGP integration (YubiKey 5 supports EdDSA with Curve25519) [1]
- FIDO/U2F *ssh-keygen* signatures over files with *ed25519-sk* [2]
- Using *signify* [6] with YubiKey HSM 2 [3]

Even though these solutions exists, we currently use signify (actually: *minisign* [7]) for signing our software releases and documents. The reason for this is that we can use standard tooling available on multiple platforms and that verifying the signatures is very easy to implement (on mobile). Unfortunately there is no “ready to use” implementation on affordable hardware of *signify*/*minisign* using a hardware token. An issue exists talking about implementing hardware support [5].

The signify/minisign approach is interesting because it seems to get some traction in some projects. OpenBSD has been using signify for a long time, GrapheneOS (an Android distribution focusing on security/privacy) also uses it. We are not aware of any project using the SSH signature.

Research / Scope

The project could extend in different directions: a literature study / experimentation evaluating all possible solutions to this problem and judge them based on the requirements. It could also go into an implementation that can match all requirements. Ideally this research is not done in isolation, but also by interacting with other potential interested parties of such work, e.g. the Debian project, GrapheneOS, SoloKey, minisign to come to a comprehensive design/implementation that is usable by as many projects as possible without compromising simplicity and security.

Deliverables

An overview of all possible solutions, not limited to the ones already found, but there could be more, that match (some of) the requirements would be part of this. A requirements gathering / evaluation of the of other projects for such hardware based solution to see if there is any overlap, or overlap can be found if there is interest in this kind of solutions by the other projects.

Resources

[1] https://developers.yubico.com/PGP/YubiKey_5.2.3_Enhancements_to_OpenPGP_3.4.html
[2] https://undeadly.org/cgi?action=article;sid=20201016053038
[3] https://marc.info/?t=155723344400002&r=1&w=2 (see entire 3 message thread)
[4] https://wiki.debian.org/Teams/Apt/Spec/AptSign
[5] https://github.com/jedisct1/minisign/issues/100
[6] https://man.openbsd.org/signify
[7] https://jedisct1.github.io/minisign/
[8] https://latacora.singles/2019/07/16/the-pgp-problem.html