Asfaload: securing downloads from the internet

Bauduin Raphaël

www.asfaload.com

2025-02-01

Problem statement

Downloads from the internet

We download a lot of software from the internet (Github Releases, Docker images, ...)

Problem statement

Downloads from the internet

We download a lot of software from the internet (Github Releases, Docker images, . . .)

No authenticity checks

But authenticity check is often impossible (checksums are not authenticity checks)

A problem causing the problem

We need a solution for median humans

Existing authentication system are not usable by mere mortals

Causes of the problem

- Signature schemes hard to use
  - few people use GPG
  - key management stumbling block

Causes of the problem

- Signature schemes hard to use
    - few people use GPG
    - key management stumbling block
- Sha sums insufficient for authenticity checks
    - sha sums only ensure integrity
    - no security benefits downloading checkums from the same source

Asfaload proposal

Propose an **easy to use**, **multi-sig**, file **authentication** system to secure downloads from the internet

Usability goals

- **Only the URL of the file to be downloaded is needed**
  - No signature or other bundle needed to securely download
- **No public key import**
  - would trust paths be checked by end users?

Design decisions

- **Use existing building blocks as much as possible**
  - git, minisign (based on OpenBSD signify)
  - Base our solution on checksums files
    - often already published by projects
    - → sign checksums file

Design decisions

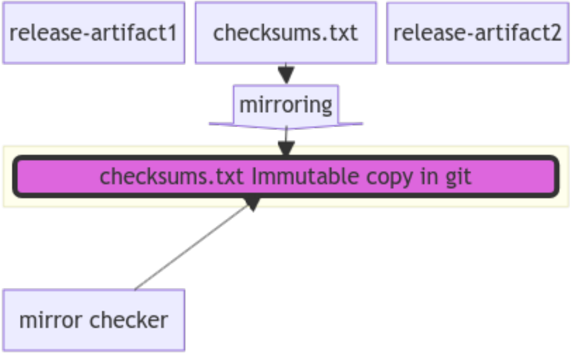
- **Use existing building blocks as much as possible**
  - git, minisign (based on OpenBSD signify)
  - Base our solution on checksums files
    - often already published by projects
    - → sign checksums file
- **Maintain an append-only mirror of checksum files**
  - Introduces an index file
    - Required as no standard naming of checksums files
  - Even without file-signing, can increase security
  - git repo, auditable

- **Authenticate at publishing repo level**
  - Inspiration from Let’s Encrypt
  - A user proves control of the repo
  - We check a released file was published by repo owner
  - We do not use the identity of the user

- **Authenticate at publishing repo level**
  - Inspiration from Let's Encrypt
  - A user proves control of the repo
  - We check a released file was published by repo owner
  - We do not use the identity of the user
- **Introduce easy to use multi signature scheme for publishers**
  - Helps in case publishing account is breached

- **Authenticate at publishing repo level**
  - Inspiration from Let's Encrypt
  - A user proves control of the repo
  - We check a released file was published by repo owner
  - We do not use the identity of the user
- **Introduce easy to use multi signature scheme for publishers**
  - Helps in case publishing account is breached
- **Be publication platform agnostic**
  - Though we start with Github
  - Will support other services like Gitlab, Forgejo, self-hosted, . . .

Asfaload Mirror



- mirror: https://github.com/asfaload/checksums
- mirror-checker:
  - code: https://github.com/asfaload/mirror-checker
  - instance: https://mirror-checker.taktiki.com/. Run one yourself!

Workflows

Authenticate repo control

- Put signers public keys in `asfaload.signers.json` in repo root
  - mirrored
- Initial signers are trusted, subsequent changes to be signed

signers json

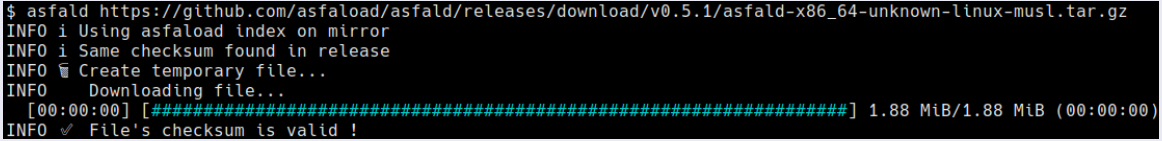
```
{
  "version": "1.0",
  "threshold": 2,
  "signers": [
      {
        "format": "minisign",
        "pubkey": "RWSNbF6ZeLYJLBOKm8a2QbbSb3U+K4ag1YJENgvRXfKEC6RqICqYF+NE"
      },
      {
        "format": "minisign",
        "pubkey": "RWTUManqs3axpHvnTGZVvmaIOOz0jaV+SAKax8uxsWHFkcnACqzL1xyv"
      },
      {
        "format": "minisign",
        "pubkey": "RWTsbRMhBd0yL8hSYo/Z4nRD6O5OvrydjXWyvd8W7Q0TftBOKSSn3PH3"
      }
    ]
}
```

Publishing a release

- Include a checksums file in the release
- Checksums are mirrored
- Signers need to sign checksums on the mirror
  - if the file is legit, accept the publication and sign checksums file
  - if the publication was not expected, it can still be blocked
- We will provide tools to facilitate this
  - but it could probably be done by Pull Requests too [TBD]

Downloading a file

- Facilitated by the use of our tool asfald
- Signatures support ongoing
- Only the url of the downloaded file is passed to `asfald`
  - no signature bundle
  - no checksums file location
- `asfald` transparently checks the file was signed by repo owner
  - if yes, saves the file
  - if not, aborts



```
$ asfald https://github.com/asfaload/asfald/releases/download/v0.5.1/asfald-x86_64-unknown-linux-musl.tar.gz
INFO i Using asfaload index on mirror
INFO i Same checksum found in release
INFO 🗑 Create temporary file...
INFO    Downloading file...
  [00:00:00] [####################################################################] 1.88 MiB/1.88 MiB (00:00:00)
INFO ✓  File's checksum is valid !
```

Figure 1: Asfald example output

Updating signers

- Valid signers are found on the mirror
- Every change to signers needs to be signed by valid signers and new signers
    - once adequatly signed, it is updated on the mirror

Implementation

Status

- Checksums mirror operational
  - https://github.com/asfaload/checksums
  - mirror checker: https://github.com/asfaload/mirror-checker/
- `asfald` available
  - https://github.com/asfaload/asfald
  - uses mirror
    - already improves security
- signatures in development, soon to be available
  - validated in a Proof of Concept
  - looking for testers!

Get in touch!

- https://www.asfaload.com
- https://github.com/asfaload
- https://github.com/asfaload/spec
- https://mastodon.social/@asfaload