The Hitchhiker's Guide to Online Anonymity

How to check files for safety/integrity and authenticity:

The PDF and ODT files of this guide are cryptographically signed using GPG and Minisign. Their integrity can be verified with the published SHA256 Checksum hashes on this website. SHA256 checksums of all the PDF and ODT files are available here in the `sha256sum.txt` file. SHA256 checksums, signatures, and VirusTotal ("VT") checks of the releases files (containing the whole repository) are available within the latest release information at `https://github.com/Anon-Planet/thgtoa/releases/latest` which will be available as soon as we have a stable release.

The GPG signatures for each PDF and ODT files are available here: - PDF (Light Theme) Main and Mirrors: guide.pdf.asc - ODT Main and Mirrors: guide.odt.asc

The Minisign signatures for each PDF and ODT files are available here: - PDF (Light Theme) Main and Mirrors: guide.pdf.minisig - ODT Main and Mirrors: guide.odt.minisig

How to check the integrity of files using SHA256 checksums:

First get the hash of your local file by following these steps for your OS:

Windows: - From a command prompt, run `certutil -hashfile filename.txt sha256` - Compare the obtained hash result of your local file to the online file's published hash. They should match.

macOS: - From a terminal, run `shasum -a 256 /full/path/to/your/file` - Compare the obtained hash result of your local file to the online file's published hash. They should match.

Linux: - From a terminal, run `sha256sum /full/path/to/your/file` - Compare the obtained hash result of your local file to the online file's published hash. They should match.

All commits and releases on this repository are cryptographically signed and verified by each collaborator (check for the "Verified" tags on commits and releases).

How to verify the the authenticity and integrity of files using GPG:

To verify files with GPG signatures, you should first install gpg on your system: - Windows: Install gpg4win from `https://www.gpg4win.org/download.html` -

MacOS: Install GPG Tools from `https://gpgtools.org/` - Linux: gpg should be installed by default. If not, use your Linux package manager to install it such as apt (debian) or rpm (red hat).

Import the master signing key from a trusted source of the publisher using the following command from a command prompt or terminal:

```
gpg --auto-key-locate nodefault,wkd --locate-keys 9EA98278639F1CD853E096CBFF94507587
```

In theory this command should fetch the key from the a default pool server. If this doesn't work, you can also download/view it directly from here (in our case): `https://anonymousplanet.org/pgp/AnonymousPlanet-Master-Signing-Key_9EA98278639F1CD853E096CBFF94507587A6A9B9.asc`

As well as the published key on any keyserver below (search for the fingerprint `9EA98278639F1CD853E096CBFF94507587A6A9B9`): - `https://pgp.mit.edu` - `https://keys.openpgp.org` - `https://keyserver.ubuntu.com`

You should then import it manually by issuing the following command on any OS:

```
gpg --import 9EA98278639F1CD853E096CBFF94507587A6A9B9.asc
```

The master signing key allows you to verify all other project-related keys. Once you have the master signing key and are confident it's the correct key (nobody has tampered with it), mark the key as trusted by locally signing it:

```
gpg --lsign-key 9EA98278639F1CD853E096CBFF94507587A6A9B9
```

Alternatively, if you use Kleopatra, it will ask you to certify the key. Certify the key to mark it as trusted.

Once you have the master key downloaded, imported, and certified, you will obtain a copy of the release key.

`gpg --auto-key-locate nodefault,wkd --locate-keys 83A6CF9EF57AC25B5C7F5D29285E6048A1` (to import the release signing key)

`https://anonymousplanet.org/pgp/AnonymousPlanet-Release-Signing-Key_83A6CF9EF57AC25B5C7F5D29285E6048A12321B2.asc` (to download the key yourself)

If you use GPG directly, you won't need to mark the release signing key as trusted, because it's already signed by the master signing key. If you use Kleopatra, the process to import the release signing key is the same as importing the master signing key.

Finally, verify the asc signature file (links above) against the PDF file by issuing the following example command:

```
gpg --verify guide.pdf.asc guide.pdf
```

This should output a result showing it matches a signature created by the release signing key, and is therefore a good result.

How to verify the the authenticity and integrity of the files using Minisign:

To verify the files with Minisign:

- First, download minisign from `https://jedisct1.github.io/minisign/`.
- Download the files along with their *.minisig signature file (these should be in the same directory).
- Download the Minisign public key available on the website and repository: `minisign.pub` (again, place it in the same directory for convenience).
- Run the following command in a command prompt or terminal within the directory with both files: `minisign -Vm guide.pdf -p minisign.pub`.
- Output should show `Signature and comment signature verified`.

How to check the relative safety of files or even URLs (such as https://anonymousplanet.org) using VirusTotal:

Note: we do not endorse VirusTotal. It should be used with extreme caution, never with any sensitive files, due to their privacy policies. Do not upload sensitive files to VirusTotal.

The PDF and ODT files of this guide have been automatically scanned by VT, see the links below for an example but do not trust these hashes blindly. Check the hashes match and re-upload to VT if needed: - PDF file: [VT Scan] - ODT file: [VT Scan]

Additional manual safety checks for the PDF files:

For additional safety, you can always double check the PDF files using the PDFID tool which you can download at `https://blog.didierstevens.com/programs/pdf-tools/`. (You might be wondering: "Why should I trust a random python script?" Well, it is open-source and well-known. It is also probably a safer bet than trusting a random PDF).

Here are the steps:

- Install the latest version (e.g., 3.10.6 stable) of Python, download pdfid and, from a command prompt or terminal, run:

```
python pdfid.py file-to-check.pdf
```

And you should see the following entries at **o** for safety, this o means there is no Javascript or any action that could possibly execute malicious macros, scripts, etc. Normally this won't be necessary as most modern PDF readers won't execute those scripts anyway.

```
/JS                    0 #This indicates the presence of Javascript
which could be malicious
/JavaScript            0 #This indicates the presence of Javascript
which could be malicious
/AA                    0 #This indicates the presence of automatic action
on opening
/OpenAction            0 #This indicates the presence of automatic action
on opening
/AcroForm              0 #This indicates the presence of AcroForm which
could contain malicious JavaScript
/JBIG2Decode           0 #This indicates the PDF uses JBIG2 compression
which could be used for obfuscating malicious content
/RichMedia             0 #This indicates the presence rich media within
the PDF such as Flash
/Launch                0 #This counts the launch actions
/EmbeddedFile          0 #This indicates there are embedded files within
the PDF
/XFA                   0 #This indicates the presence of XML Forms within
the PDF
```