

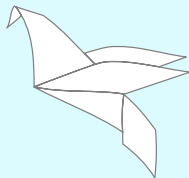
Sequoia PGP

Rethinking OpenPGP Tooling

Neal H. Walfield <neal@sequoia-gpg.org>
8F17777118A33DDA9BA48E62AACB3243630052D9

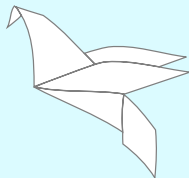
FOSDEM 2024

February 4, 2024




Outline

- Introduction
- Design and Implementation
- Day-to-Day Usage
















What is Sequoia PGP?

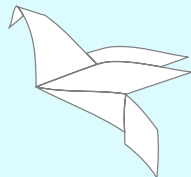
S sequoia-pgp  ⏶ New subgroup New project ⋮

A free implementation of OpenPGP in Rust.

Subgroups and projects Shared projects Archived projects Updated ⏴ ⏵

 S sequoia-sq 	★ 3	21 hours ago
 S Sequoia Web of Trust 	★ 3	21 hours ago
 S Sequoia Cert Store 	★ 0	22 hours ago
  sequoia  Maintainer	★ 380	3 days ago
 S sequoia-keystore 	★ 0	4 days ago
 O OpenPGP interoperability test suite 	★ 3	5 days ago

- An OpenPGP implementation
- Services
- Tooling
- Applications
- Paradigm Shift



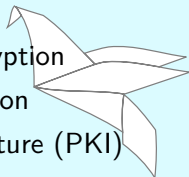
What is OpenPGP?

```
draft-ietf-openpgp-crypto-refresh-13


Network Working Group                                P. Wouters, Ed.
Internet-Draft                                       Aiven
Obsoletes: 4880, 5581, 6637 (if approved)           D. Huigens
Intended status: Standards Track                    Proton AG
Expires: 7 July 2024                                J. Winter
                                                    Sequoia-PGP
                                                    Y. Niibe
                                                    FSIJ
                                                    4 January 2024

                                OpenPGP
draft-ietf-openpgp-crypto-refresh-13
```

- IETF standard
 - Derived from PGP
 - First version: 1996
 - Next version: 2024
- Interchange format
 - Messages
 - Certificates
- Encryption and Decryption
- Signing and Verification
- Public Key Infrastructure (PKI)
















What is Sequoia PGP?

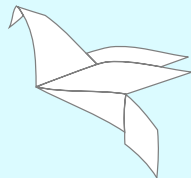
S sequoia-pgp  🔍 New subgroup New project ⋮

A free implementation of OpenPGP in Rust.

Subgroups and projects Shared projects Archived projects Updated ⌵

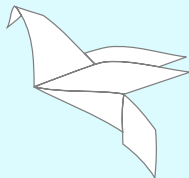
 S sequoia-sq 	★ 3	21 hours ago
 S Sequoia Web of Trust 	★ 3	21 hours ago
 S Sequoia Cert Store 	★ 0	22 hours ago
  sequoia  Maintainer	★ 380	3 days ago
 S sequoia-keystore 	★ 0	4 days ago
 O OpenPGP interoperability test suite 	★ 3	5 days ago

- An OpenPGP implementation
- Services
- Tooling
- Applications
- Paradigm Shift



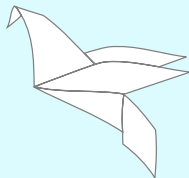
Sequoia's Technical Goals

- Library-first architecture
- Unopinionated low-level interfaces
- Safe by default
- Gradual, opinionated higher-level interfaces
- Optional services



Motivation: The Negative

- Complaints heard from *some* GnuPG users:
 - CLI hard to use
 - CLI-first approach is brittle
 - APIs are too opinionated
 - Services shouldn't be mandatory
 - Poor scalability



Motivation: The Positive

- “We use a lot of different encryption technologies, but probably none more important than GPG.” – Alex Abdo, ACLU (2017)



Sze Ming, Sinar Project



Cindy Cohn, EFF



Alex Abdo, ACLU



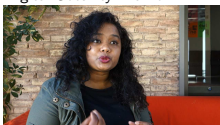
Seamus Tuohy,
Digital Security Trainer



Michał 'Rysiek' Woźniak,
OCCRP



Jason Reich,
BuzzFeed News



Thenmozhi Soundararajan
Equality Labs



Matt Mitchell,
CryptoHarlem

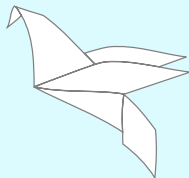


Benjamin Ismail,
Reporters without Borders

GnuPG Stories (2017), https://www.youtube.com/playlist?list=PLjX3x3GHo0Wks-VCjFBu_Yk5l1-l9mJzi

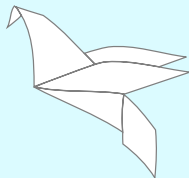
Sequoia's Pre-history

- Project started in 2017
- By three former GnuPG developers
 - Justus Winter, Kai Michaelis, Neal H. Walfield
- Worked on gpg, supported application developers, talked to users
- Had ideas on how to change GnuPG



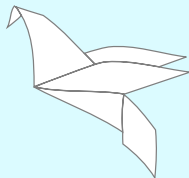
A GnuPG Evolution or Revolution?

- Many technical discussions with Werner Koch
- No significant convergence of visions



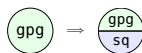
Resolving the Conflict

- Continue the established approach?
- Pursue the “Sequoia” vision?

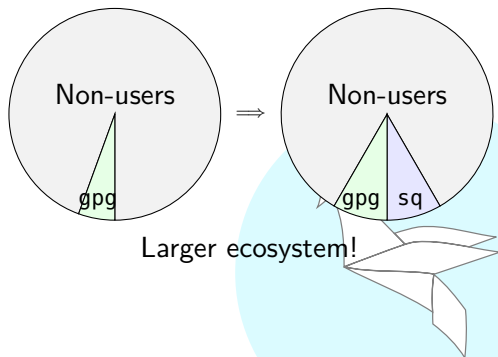


Not One, Both

- Resolution: part ways
- More choice for users
 - Diversity of needs
 - Win over non-users
- Interoperable protocol
 - Network effects help other implementations
- Ecosystem wins!
- **Privacy and security win!**



Split users?



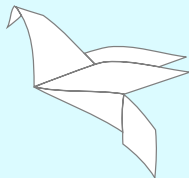
Larger ecosystem!

An Ode to Werner

- Sequoia owes its existence to Werner
- Inspiration to make GnuPG better
- Inspiration to work on cryptography
- Inspiration to defend privacy
- If Justus, Kai, and I are Sequoia's parents, then Werner is Sequoia's spiritual grandfather



Olive Branch, Mislav Marohnić
CC BY 2.0 Deed

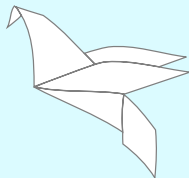


An Ode to Werner

- Sequoia owes its existence to Werner
- Inspiration to make GnuPG better
- Inspiration to work on cryptography
- Inspiration to defend privacy
- If Justus, Kai, and I are Sequoia's parents, then Werner is Sequoia's spiritual grandfather

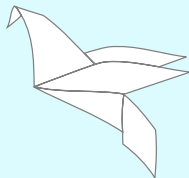


Olive Branch, Mislav Marohnić
CC BY 2.0 Deed



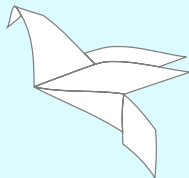
Not Both, Many

- GnuPG (C)
- GopenPGP (Go)
- OpenPGP.js (JavaScript)
- PGPainless (Java)
- PGPpy (Python)
- RNP (C++)
- rPGP (Rust)
- Sequoia (Rust)



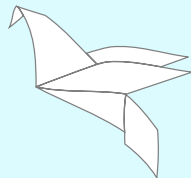
Ensuring Interoperability

- Interoperability is important
 - Prevents vendor lock-in
 - Network effects *for all*
- A standard is not enough
- OpenPGP Interoperability Test Suite
 - 131 Tests
 - 1510 Test Vectors

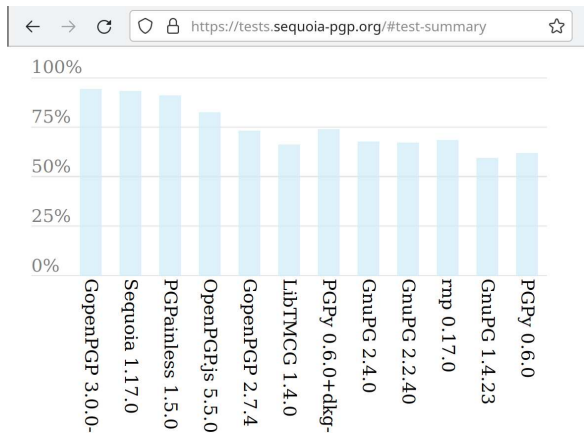


Ensuring Interoperability

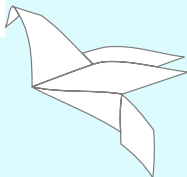
- Interoperability is important
 - Prevents vendor lock-in
 - Network effects *for all*
- A standard is not enough
- OpenPGP Interoperability Test Suite
 - 131 Tests
 - 1510 Test Vectors



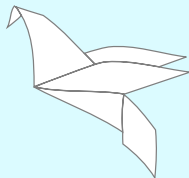
Interoperability Test Suite




- Most implementations tested
- rPGP support being added by Heiko Schäfer 🧑

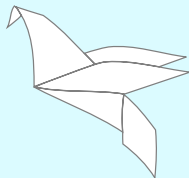


- Introduction
- Design and Implementation
- Day-to-Day Usage

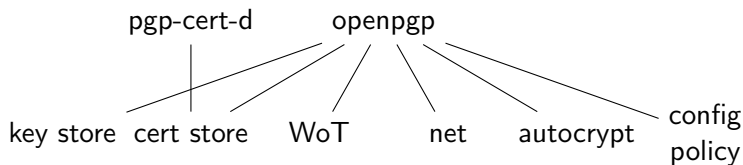


Sequoia's Architecture

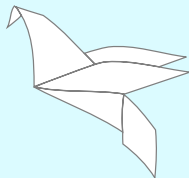
- Library-first approach
 - Applications built on library
 - CLI is less powerful than the library
- High-level components are optional
- Services as daemons *or* co-located
 - Daemon
 - Process separation: Avoid heartbleed 
 - Multiplex resources
 - Share state
 - Co-located
 - Restricted environment
 - Fallback to increase robustness



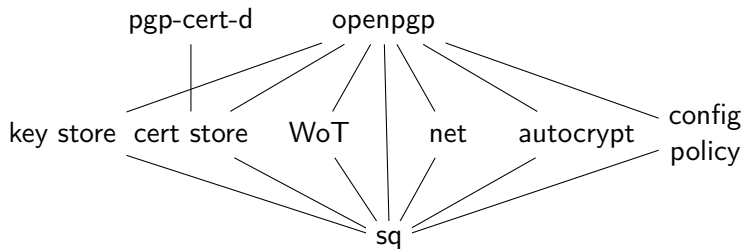
Components



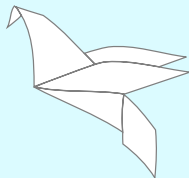
- openpgp: The low-level library
- key store: Private key operations
- pgp-cert-d: On-disk certificate store
- cert store: In-memory certificate store
- WoT: Web of trust engine
- net: Key server, WKD, and DANE support
- autocrypt: Autocrypt functionality
- config policy: Reads and parses a cryptographic policy

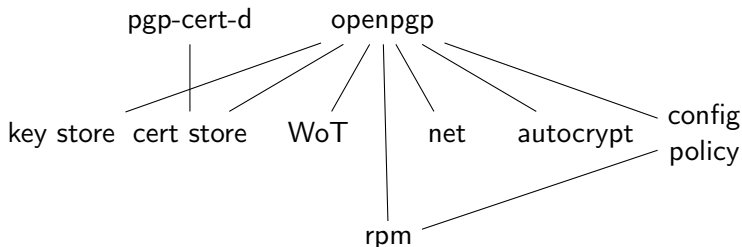


Components

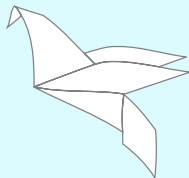


- sq
 - Uses all high-level libraries and services

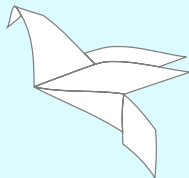




- RPM Package Manager (rpm)
 - Doesn't use secret key material
 - Has its own certificate store
 - Has its own trust model
 - Uses the common policy configuration



- Unopinionated low-level APIs, safe by default
- Opinionated high-level APIs, built on low-level APIs

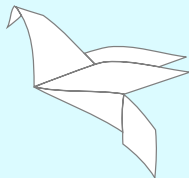


Serializes a certificate, strips secret key material by default:

```
cert.serialize(&mut output)?;
```

To serializes the secret key material, we have to opt-in:

```
cert.as_tsk().serialize(&mut output)?;
```



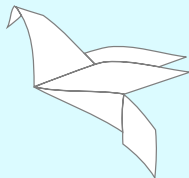
Progressive High-Level API

Creates a certificate:

```
let (cert, rev) = CertBuilder::general_purpose(None, None)
    .add_userid("Alice <alice@example.org>")
    .generate()?;
```

Creates a certificate with a decentralized social proof:

```
let template = SignatureBuilder::new(SignatureType::CasualCertification)
    .set_notation("proof@metacode.biz", b"https://mastodon.example/@alice",
        NotationDataFlags::empty().set_human_readable(),
        false)?;
let (cert, rev) = CertBuilder::general_purpose(None, None)
    .add_user_id_with("Alice <alice@example.org>", template)?
    .generate()?;
```



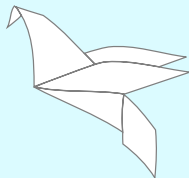
Progressive High-Level API

Creates a certificate:

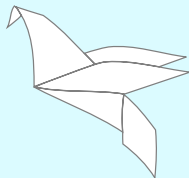
```
let (cert, rev) = CertBuilder::general_purpose(None, None)
    .add_userid("Alice <alice@example.org>")
    .generate()?;
```

Creates a certificate with a decentralized social proof:

```
let template = SignatureBuilder::new(SignatureType::CasualCertification)
    .set_notation("proof@metacode.biz", b"https://mastodon.example/@alice",
        NotationDataFlags::empty().set_human_readable(),
        false)?;
let (cert, rev) = CertBuilder::general_purpose(None, None)
    .add_user_id_with("Alice <alice@example.org>", template)?
    .generate()?;
```



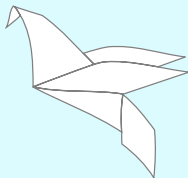
- Introduction
- Design and Implementation
- Day-to-Day Usage



- sq: Sequoia's primary CLI
- Subcommand-style interface

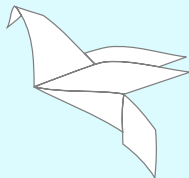
```
$ sq encrypt --recipient-email neal@sequoia-pgp.org
```
- Clear separation of options

```
$ sq sign --recipient-email neal@sequoia-pgp.org
error: unexpected argument '--recipient-email' found
```
- Consistent usage between subcommands
 - --email option has same semantics across subcommands



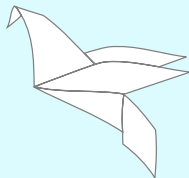
Away from Curated Keyrings

- Local certificates are mostly assumed to be authenticated
- Say yes to get work done:
\$ **gpg -e -r dkg@debian.org**
gpg: 0x38024D718ABA3F3B: There is no assurance this key belongs to the named user
...
Use this key anyway? (y/N)
- Certifying user IDs is tiresome



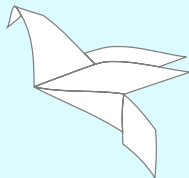
Towards Strong Authentication

- Local certificate store is just a cache
- Self-signed user IDs are just a hint
- Certificates can only be addressed by **authenticated** IDs
- Embrace the web of trust
- Is this a usability nightmare?
- Let's see...



Towards Strong Authentication

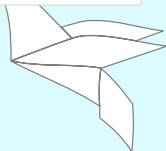
- Local certificate store is just a cache
- Self-signed user IDs are just a hint
- Certificates can only be addressed by **authenticated** IDs
- Embrace the web of trust
- Is this a usability nightmare?
- Let's see...



What is Authentication?

- What certificate should I use for Alice?
- Who does the certificate BB7E9101495E6BF7 belong to?
- Self-signatures are useless for authentication!
- Is this Alice's or Mallory's certificate?

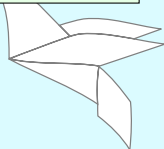
Primary Key (Fingerprint)
Encryption Key
Binding Signature
Signing Key
Binding Signature
Alice <alice@example.org>
Binding Signature
Alice <alice@other.org>
Binding Signature



What is Authentication?

- What certificate should I use for Alice?
- Who does the certificate BB7E9101495E6BF7 belong to?
- Self-signatures are useless for authentication!
- Is this Alice's or Mallory's certificate?

Primary Key (Fingerprint)
Encryption Key
Binding Signature
Signing Key
Binding Signature
Alice <alice@example.org>
Binding Signature
Alice <alice@other.org>
Binding Signature



Google Security Blog

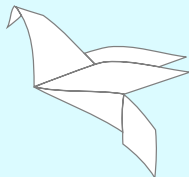
Chrome's Plan to Distrust Symantec Certificates

September 11, 2017

Posted by Devon O'Brien, Ryan Sleevi, Andrew Whalley, Chrome Security

At the end of July, [the Chrome team and the PKI community converged upon a plan to reduce, and ultimately remove, trust in Symantec's infrastructure](#) in order to uphold

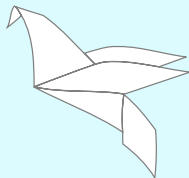
- Centralized: easy, but unsafe
 - X.509: Hundreds of centralized CAs
 - Any one can trick you
 - Certificate transparency helps
 - Signal
 - One key server
 - On same infrastructure as the message transport
 - But, can trust Signal Foundation
- Peer to peer: safe, but high upfront overhead
 - Check fingerprints or safety numbers
- Consistency: easy until you have a problem
 - Trust on First Use (TOFU)



Good Enough for Most?

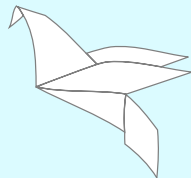
The test of a civilization is the way that it cares for its helpless members.

Pearl Buck, My Several Worlds



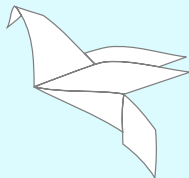
Good Enough for Most?

- Goal: a progressive system that serves a range of needs
- Approach
 - Provide a range of tools to increase confidence
 - Support user
 - Have a knob to set a threshold based on the threat model

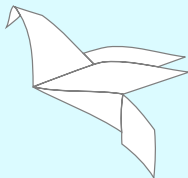


Web of Trust: A Powerful, Flexible PKI

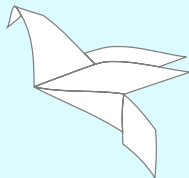
- Everyone can act like a certification authority
- Users have their own personal trust roots
- Can use weak evidence
- Can *combine* evidence
- Modes of operation
 - ✓ Centralized
 - ✓ Federated
 - ✓ Peer to peer



- If the web of trust is so good, why has it not succeeded yet?
- Missing tools to:
 - Automatically incorporate evidence into a web of trust
 - Easily manage a web of trust
- Until now. . .

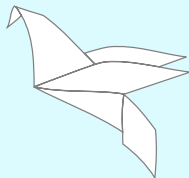


- If the web of trust is so good, why has it not succeeded yet?
- Missing tools to:
 - Automatically incorporate evidence into a web of trust
 - Easily manage a web of trust
- Until now. . .



Example: Encrypting a Message

- Task: encrypt a message to Daniel Kahn Gilmor <dkg@debian.org>

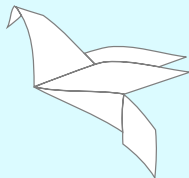


Let's Just Try It

```
$ sq encrypt --recipient-email dkg@debian.org  
Error: --recipient-email
```

Caused by:

```
No certificates are associated with "dkg@debian.org"
```



Getting Daniel's Certificate

```
$ sq network fetch dkg@debian.org
```

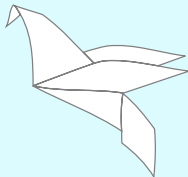
Importing 4 certificates into the certificate store:

1. 0EE5BE979282D80B9F7540F1CCD2ED94D21739E9 Daniel Kahn Gillmor
<dkg@fifthhorseman.net>
2. C29F8A0C01F35E34D816AA5CE092EB3A5CA10DBA Daniel Kahn Gillmor
3. C4BC2DDB38CCE96485EBE9C2F20691179038E5C6 Daniel Kahn Gillmor
<dkg@debian.org>
4. D477040C70C2156A5C298549BB7E9101495E6BF7 Daniel Kahn Gillmor

Imported 4 new certificates, updated 0 certificates, 0 certificates unchanged, 0 errors.

After checking that a certificate really belongs to the stated owner, you can mark the certificate as authenticated using:

```
sq pki link add FINGERPRINT
```



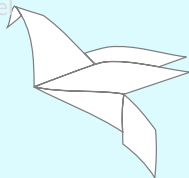
Getting Daniel's Certificate

```
$ sq network fetch dkg@debian.org
```

Importing 4 certificates into the certificate store:

...

- sq network fetch found 4 certificates
- Which one is the right one?
- Did sq network fetch even find the right one?
- Best: Ask Daniel
- Good: Ask someone who knows Daniel's certificate
- Better: *Ask multiple entities*, combine evidence
 - Weigh evidence according to entity's reliability
 - Amount of needed evidence depends on the threat model



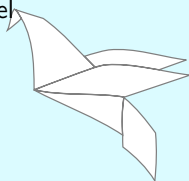
Getting Daniel's Certificate

```
$ sq network fetch dkg@debian.org
```

Importing 4 certificates into the certificate store:

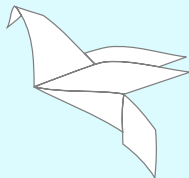
...

- sq network fetch found 4 certificates
- Which one is the right one?
- Did sq network fetch even find the right one?
- Best: Ask Daniel
- Good: Ask someone who knows Daniel's certificate
- Better: *Ask multiple entities*, combine evidence
 - Weigh evidence according to entity's reliability
 - Amount of needed evidence depends on the threat model



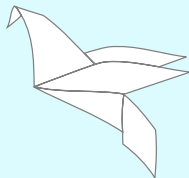
Rudimentary Evidence

- Public directories
 - Key servers
 - `keys.openpgp.org`: Validating key server
 - `keys.mailvelope.com`: Validating key server
 - `proton.me`: Validating key server for proton users
 - SKS: Free for all
 - WKD: Entry set by user / admin
 - DANE: Entry set by user / admin
- `sq network` fetch queries all of them!
- ... and records the evidence!



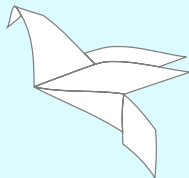
Rudimentary Evidence

- Public directories
 - Key servers
 - `keys.openpgp.org`: Validating key server
 - `keys.mailvelope.com`: Validating key server
 - `proton.me`: Validating key server for proton users
 - SKS: Free for all
 - WKD: Entry set by user / admin
 - DANE: Entry set by user / admin
- `sq network` fetch queries all of them!
- ...and records the evidence!

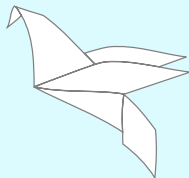


Rudimentary Evidence

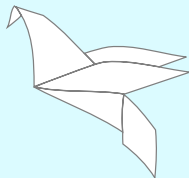
- Public directories
 - Key servers
 - `keys.openpgp.org`: Validating key server
 - `keys.mailvelope.com`: Validating key server
 - `proton.me`: Validating key server for proton users
 - SKS: Free for all
 - WKD: Entry set by user / admin
 - DANE: Entry set by user / admin
- sq network fetch queries all of them!
- ...and records the evidence!



- Evidence stored as web of trust data structures
 - Creates a *Shadow CA* for entities with > 0 reliability
 - `keys.openpgp.org`: Yes, it validates user IDs.
 - SKS: No, anyone can upload a certificate for `dkg@debian.org`
 - Shadow CA certifies the returned user IDs and certificate pairs.
- Evidence is automatically combined by web of trust engine



- Trust root and shadow CAs created automatically
- Shadow CAs trusted minimally by default
- Trust root, shadow CAs, and certificates are marked as unexportable
 - Protect user's privacy



Examining the Evidence

```
$ sq pki list dkg@debian.org
```

```
[ ] D477040C70C2156A5C298549BB7E9101495E6BF7 <dkg@debian.org>:  
marginally authenticated (2%)
```

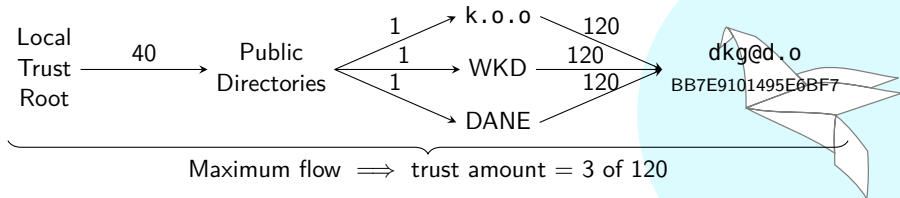
```
Path #1 of 3, trust amount 1:
```

- 788B10EF1FF13B3D07C66740F202C5759961E844 "Local Trust Root"
partially certified (amount: 40 of 120) ...
- 6E72B2F036619E5A712DB3C6FAF635227D91FC53 "Public Directories"
partially certified (amount: 1 of 120) ...
- 1CB68A1218567FB18DB97176A41F17E9C1439134 "Downloaded from keys.openpgp.org"
certified the following binding on 2024-02-01
- D477040C70C2156A5C298549BB7E9101495E6BF7 "<dkg@debian.org>"

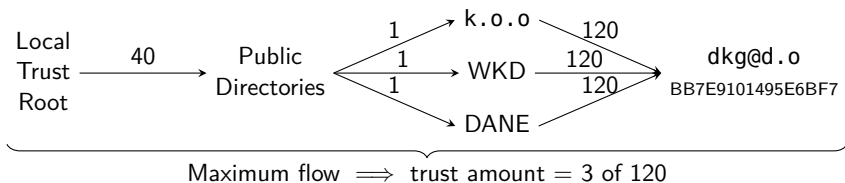
```
Path #2 of 3, trust amount 1:
```

```
...
```

```
Could not authenticate any paths.
```

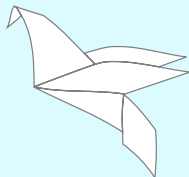


Examining the Evidence



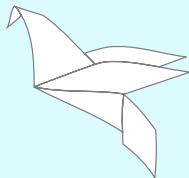
- Observations:

- Shadow CAs are partially trusted
- Public directories intermediary acts as a resistor
- Three pieces of evidence
- Binding not fully authenticated
- No evidence for other certificates



What now?

- If we are not sufficiently convinced, get more evidence
- Once convinced, two options:
 - Create a public certification
 - Create a private link (permanent or temporary)

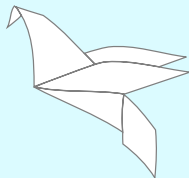


Creating a Private Link

```
$ sq pki link add D477040C70C2156A5C298549BB7E9101495E6BF7 "<dkg@debian.org>"  
Linking D477040C70C2156A5C298549BB7E9101495E6BF7 and "<dkg@debian.org>".
```

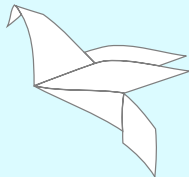
```
$ sq pki list dkg@debian.org
```

```
[✓] D477040C70C2156A5C298549BB7E9101495E6BF7 <dkg@debian.org>:  
  fully authenticated (100%)  
  ○ 788B10EF1FF13B3D07C66740F202C5759961E844 ("Local Trust Root")  
    └ certified the following binding on 2024-02-01  
      └ D477040C70C2156A5C298549BB7E9101495E6BF7 "<dkg@debian.org>"
```



Success!

```
$ sq encrypt --recipient-email dkg@debian.org  
-----BEGIN PGP MESSAGE-----  
...
```



Fully Trust keys.openpgp.org

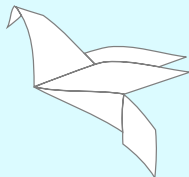
```
$ sq pki link add --ca '*' 1CB68A1218567FB18DB97176A41F17E9C1439134 --all
Linking 1CB68A1218567FB18DB97176A41F17E9C1439134 and
"Downloaded from keys.openpgp.org".
```

```
$ sq pki list dkg@fifthorseman.net
```

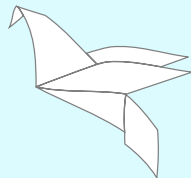
```
[✓] D477040C70C2156A5C298549BB7E9101495E6BF7 <dkg@fifthorseman.net>:
```

```
  fully authenticated (100%)
```

- 788B10EF1FF13B3D07C66740F202C5759961E844 ("Local Trust Root")
 - certified the following certificate
 - 1CB68A1218567FB18DB97176A41F17E9C1439134 ("Downloaded from keys.openpgp.org")
 - certified the following binding on 2024-02-01
 - D477040C70C2156A5C298549BB7E9101495E6BF7 "<dkg@fifthorseman.net>"

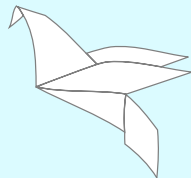


- Usage Information (TOFU)
- Monitor a URL
- Autocrypt header
- Autocrypt gossip
- Organizational CAs



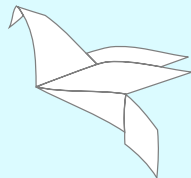
Organizational CAs

- Members of an organization delegate to a trusted *internal* entity
 - Already done in companies: IT department
 - Often done in organizations: The “Nerd”
- Bootstrap trust into an organization
 - Check one certificate, authenticate many



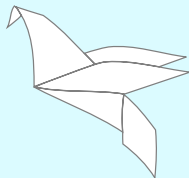
Organizational CAs

- We run a CA for sequoia-pgp.org
- You could use it as an authority for sequoia-pgp.org email addresses
- This is supported by the web of trust!
 - Trust amount: 1 to 120
 - Scope to a domain: sequoia-pgp.org



Organizational CAs

- We run a CA for sequoia-pgp.org
- You could use it as an authority for sequoia-pgp.org email addresses
- This is supported by the web of trust!
 - Trust amount: 1 to 120
 - Scope to a domain: sequoia-pgp.org

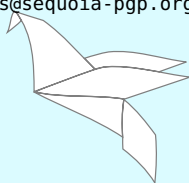


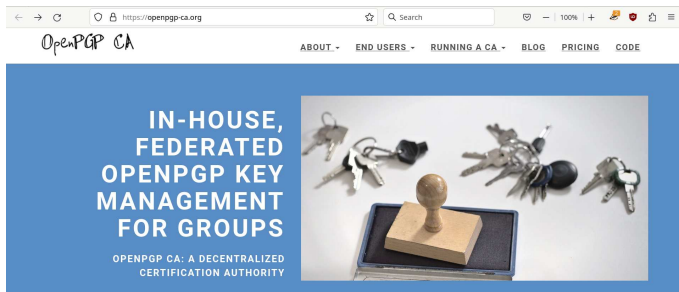
(Partially) Trusting a CA

```
$ sq pki link add --ca sequoia-pgp.org \  
> 34F9E4B6A0A70BFEC5AE45198356989DF1977575 --all  
Linking 34F9E4B6A0A70BFEC5AE45198356989DF1977575 and  
"OpenPGP CA <openpgp-ca@sequoia-pgp.org>".
```

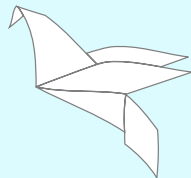
```
$ sq pki list justus@sequoia-pgp.org
```

```
[✓] CBCD8F030588653EEDD7E2659B7DD433F254904A Justus Winter <justus@sequoia-pgp.org>  
    fully authenticated (100%)  
○ 788B10EF1FF13B3D07C66740F202C5759961E844 Local Trust Root  
  ┌ certified the following certificate ...  
  └ 34F9E4B6A0A70BFEC5AE45198356989DF1977575 OpenPGP CA <openpgp-ca@sequoia-pgp.org>  
    ┌ certified the following binding on 2022-02-09  
    └ CBCD8F030588653EEDD7E2659B7DD433F254904A Justus Winter <justus@sequoia-pgp.org>  
...
```

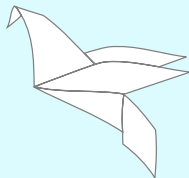




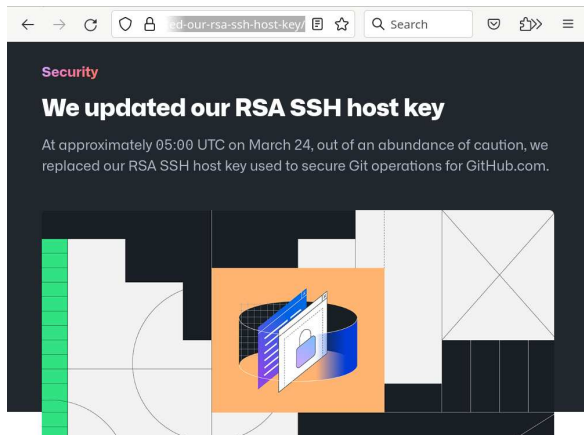
- Manage your own CA
- OpenPGP CA
- Written by Heiko Schäfer 🧑



- Authentication keys are identity keys
- If an authentication key is compromised, users have to update



Case Study: GitHub



- In 2023, GitHub's ssh private key was exposed
 - Good: Key rotated
 - Bad: All users had to update their known_hosts file

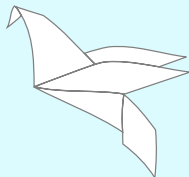


Projects > [OpenPGP-OpenSSH](#)

Improving OpenSSH's Authentication and PKI

Improving SSH Authentication with OpenPGP transitive trust

- Better approach
 - Offline identity key
 - Rotate subkey
- Project by Wiktor Kwapisiewicz and David Runge
 - ssh-openpgp-auth



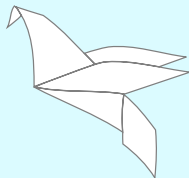


github-actions bot commented last week

```
* 739a606da6ea1e836a415a1fd7899f36b7fd5a38 Release 1.6.0.
| - Authorized by Neal H. Walfield <neal@pep.foundation> [74E445BA0E15C957]
* fc9ae07d566ff3407806d6060e23e9ab49ce69b5 Update dependencies.
| - Authorized by Neal H. Walfield <neal@pep.foundation> [74E445BA0E15C957]
* df0fed39bc9bfc0b4762088725d052485e53ede5 Don't eagerly reject expired or revoked certificates.
  - Trust root.
```

The pull request's base ([df0fed3](#)) authenticates the pull request's head ([739a606](#)).

- Signing commits means we can authenticate them
- To authenticate something, we need a policy
- sq-git defines a policy language
- sq-git checks a policy
 - Policy stored in repository: `openpgp-policy.toml`
 - Can check when pushing, when pulling, when auditing

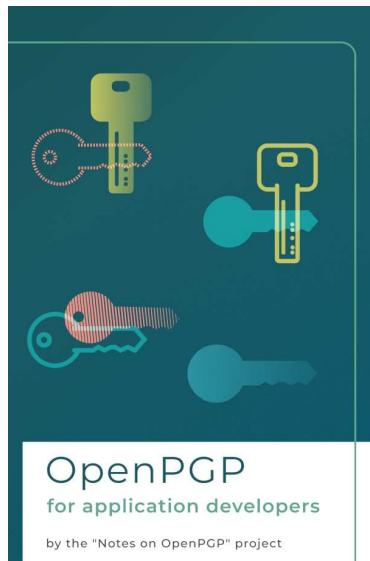


Using Sequoia Today

- sq packaged for Debian, Fedora, Arch, etc.
- gpg Chameleon
 - Implementation of gpg's interface
 - \$ **gpg --version**
gpg (GnuPG-compatible Sequoia Chameleon) 2.2.40
...
 - Uses both gpg's state and sq's
 - gpg-agent support
- Thunderbird Octopus
 - Sequoia backend for Thunderbird
 - Restores web of trust support
 - gpg-agent support

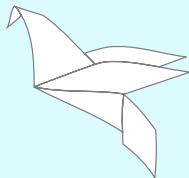


- *Notes on OpenPGP: Friendly documentation*
- By Heiko Schäfer, Paul Schaub, Ms. Uppity, Wiktor Kwapisiewicz and David Runge 🧑
- <https://openpgp.dev/>



Funding

- 2017–2023: p≡p Foundation
- 2021–now: NLNet
- 2023–2024: Sovereign Tech Fund
- Post 2024: Open Question 😊



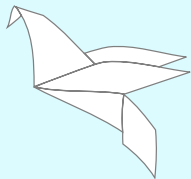
Summary

- Different users have different needs
- Sequoia
 - Different architecture
 - Different paradigms
- A diverse ecosystem is a strength
- **Winning is improving privacy and security for all!**

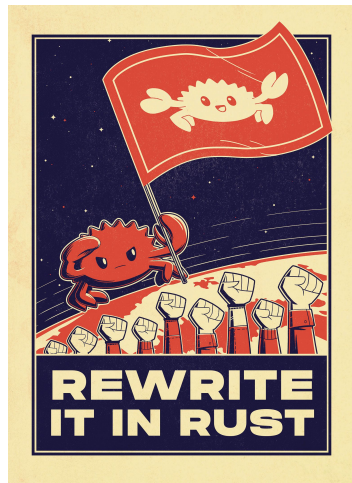
- Aside: Implement your own PKI, is the new implement your own crypto library. Don't do it.



Extra Slides

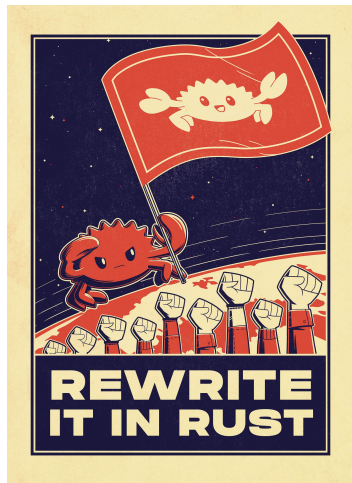


- Two easy ways to integrate Sequoia:
 - `cargo add sequoia-openpgp`
 - Rewrite It In Rust



<https://fission.codes/rewrite-in-rust/>

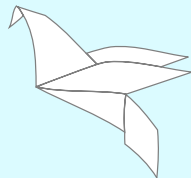
- Two easy ways to integrate Sequoia:
 - `cargo add sequoia-openpgp`
 - Rewrite It In Rust Just kidding 🤖



<https://fission.codes/rewrite-in-rust/>

Generic Bindings 🙄

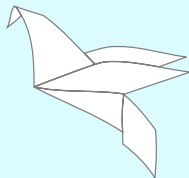
- Large, low-level API to wrap
- Hard to do in a policy-free manner
- Experience writing a C wrapper was a disaster 😱



Point Solution 👍

- Small Rust library that exports only the needed functionality
- Minimizes impedance mismatches
- Reduces language boundary crossings
- Examples:

p≡p Engine	C	3727 LOC
rpm	C	2443 LOC
SecureDrop	Python	411 LOC
Anon.io	PHP	347 LOC



A Few Users of Sequoia

- p≡p Engine
(Key management library)
- RPM Package Manager
- SecureDrop
(Whistleblower submissions)
- Anon.io
(Anonymous Email Forwarding)
- Sett (Swiss platform for exchanging medical data)
- Ripasso (Password manager)
- Qubes
- Proxmox
- Amazon
- Fortanix
- Greenbone



SECUREDROP
Share and accept documents securely.

SecureDrop is an open source whistleblower submission system that media organizations and NGOs can install to securely accept documents from anonymous sources. SecureDrop is available in **21 languages**.

[Get SecureDrop at your organization >](#)

The image shows a promotional graphic for SecureDrop. On the left is a 3D isometric cube with a keyhole on top and the letters 'SD' on the front faces. To the right of the cube is the text 'SECUREDROP' in large, bold, white letters. Below that is the tagline 'Share and accept documents securely.' and a paragraph of text describing the system. At the bottom right is a white button with the text 'Get SecureDrop at your organization >'.