A common OpenPGP Interoperability Test Suite

Justus Winter <justus@sequoia-pgp.org>

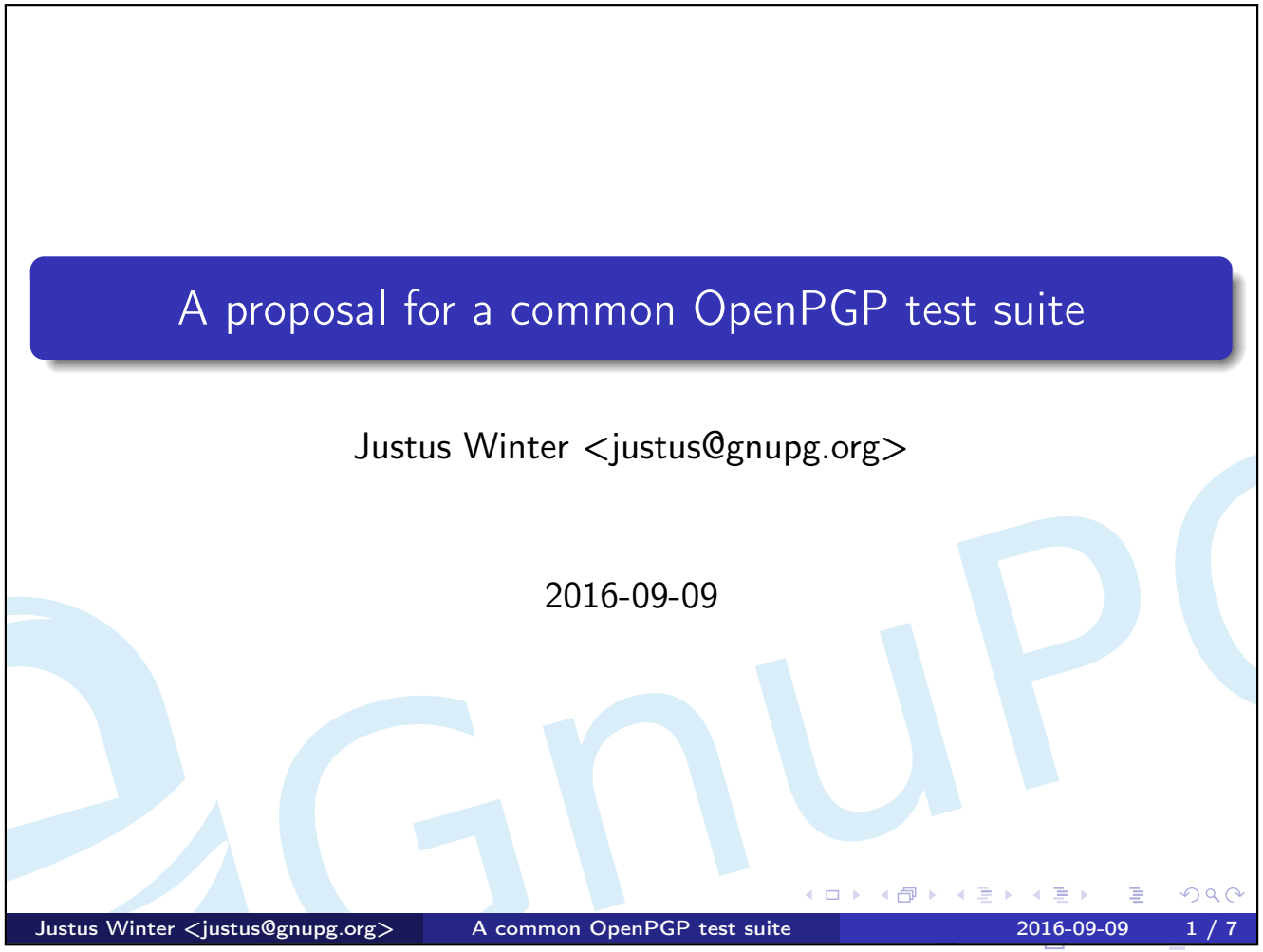
IETF 110, 2021-03-11

https://tests.sequoia-pgp.org

https://sequoia-pgp.org/talks/2021-03-ietf/openpgp-interoperability-test-suite.pdf

A little Context, please?

- OpenPGP developer for >5 years
- Nowadays employed by the pEp foundation
- An old idea whose time has come...



The Why?

Benefits. . .

- . . . for us
  - validate our implementation
  - improve the ecosystem
- . . . for users
  - better software
  - increased interoperability
- . . . for other implementations
  - free tests
- . . . for the working group
  - what's implemented
  - what's underspecified

The How?

- black box
  - consumer tests
  - producer-consumer tests
- common interface
  - Stateless OpenPGP interface

```
$ sqop generate-key >me.pgp
$ sqop encrypt me.pgp
$ sqop decrypt me.pgp
```

Example test

This is an example.

Additional artifacts:

- Certificate

Producer	Artifact	Consumer	FooPGP/1	BarPGP/2	BazPGP/3	Expectation	Comment
Base case		✓	✓	✓		✓	Interoperability concern.
Well-formed variant		✓	✓	✗		✓	Interoperability concern.
Malformed variant		✗	✓	✗		✗	Message is malformed.
Weird variant		✗	✓	✓			
Producer failure	✗					✓	Should work (TM).

An example consumer test result

About those consumer tests. . . ?

EdDSA signature encodings

OpenPGP mandates that leading zeros are stripped when encoding MPIs. This test tests whether leading zeros in S, and 0x40-prefixed R are accepted.

Additional artifacts:

- Certificate ☐

Producer	Artifact	Sequoia/1.0.0	dkg/1.2.0	GopenPGP/v2.1.1	OpenPGP.js/v4.10.10	PGPainless/CLI	RNP/0.0.0+git20210301.ffcfb63	SOPGPy/0.1.0/0.5.3	GPGME/2.2.27	GPGME/1.4.23	Expectation	Comment
MPI encoding	☐	✓	✓	✓	✓	✓	✓	✓	✓	✗	✓	MPI encoding must be supported.
S 0-padded	☐	✗	✓	✓	✓	✓	✓	✓	✓	✗		
R 0x40-prefixed	☐	✗	✗	✗	✗	✗	✗	✗	✗	✗		

And the producer-consumer tests. . . ?

Default key generation, encrypt-decrypt roundtrip

This models key generation, distribution, and encrypted message exchange. Generates a default key with the producer *P*, then extracts the certificate from the key and uses it to encrypt a message using the consumer *C*, and finally *P* to decrypt the message.

Producer	Artifact	Sequoia/1.1.0	dkg/1.2.0	GopenPGP/v2.1.1	OpenPGP.js/v4.10.10	PGPainless/CLI	RNP/0.0.0+git20210301.ffcfb63	SOPGPy/0.1.0/0.5.3	GPGME/2.2.27	GPGME/1.4.23	Expectation	Comment
Sequoia/1.1.0	☐	✓	✓	✓	✓	✓	✓	✓	✓	✗	✓	Interoperability concern.
dkg/1.2.0	☐	✗	✓	✗	✓	✓	✓	✗	✓	✓	✓	Interoperability concern.
GopenPGP/v2.1.1	☐	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	Interoperability concern.
OpenPGP.js/v4.10.10	☐	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	Interoperability concern.
PGPainless/CLI	☐	✓	✓	✓	✓	✓	✓	✓	✓	✗	✓	Interoperability concern.
RNP/0.0.0+git20210301.ffcfb63	☐	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	Interoperability concern.
SOPGPy/0.1.0/0.5.3	☐	✓	✓	✓	✓	✓	✓	✓	✓	✗	✓	Interoperability concern.
GPGME/2.2.27	☐	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	Interoperability concern.
GPGME/1.4.23	✗										✓	Interoperability concern.

Any Results?

- circa 80 tests
- around 412 test vectors
- found at least 78 bugs in 9 implementations

- improved implementations
- improved our understanding of the ecosystem
- highlights areas where implementations lack guidance

The Good, the Bad, the Ugly?

- good: algorithm support
- bad:
  - subpackets 1, 2
  - timestamps 1, 2, 3
  - unknown packets 1, 2, 3
  - expirations 1, 2
  - revocations 1, 2
  - robustness 1
  - ASCII Armor 1, 2
- ugly:
  - weak algorithms 1, 2, 3, 4

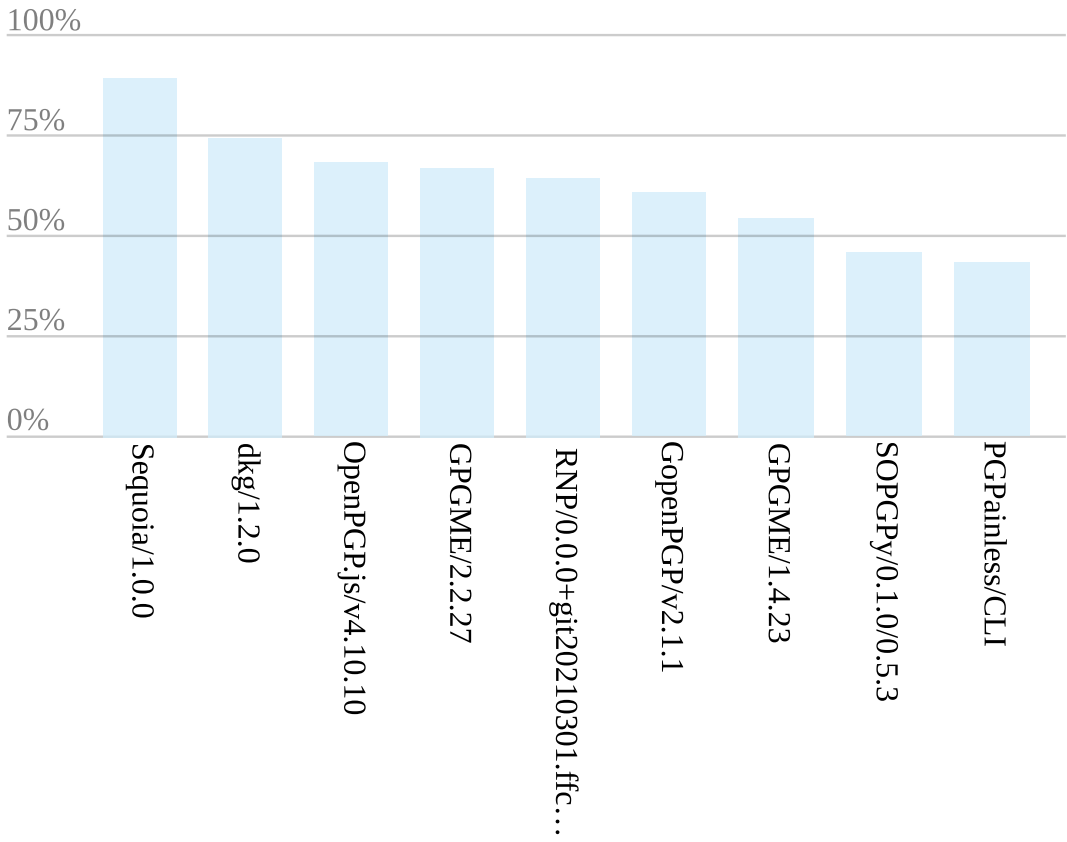


Figure: Percent of tests where an implementation agrees with the expectations on all test vectors.

Join the Fun?

- add tests
  - talk to me
  - open an issue
- add an implementation
  - AWESOME!
  - implement the Stateless OpenPGP interface
  - talk to me
- argue semantics
  - talk to me
  - open an issue
  - discuss on openpgp@ietf.org

run test suite

```
$ git clone
https://gitlab.com/sequoia-pgp/
openpgp-interoperability-test-suite
$ less README.md # optional; YOLO
$ apt install sqop # optional
$ cp config.json.dist config.json
$ editor config.json
$ cargo run -- --html-out results.html
```