Wounded hero revived

Lessons learned from porting M2Crypto to Py3k

Author: Matěj Cepl <mcepl@cepl.eu>

URL: https://matej.ceplovi.cz/clanky/PyCon18-m2crypto/slides.html

Origins

- Mitch Kapor sold Lotus to IBM and decided to perpetrate good. One of his projects was Chandler.
- Project is gone The only remainder of the project is M2Crypto, full Python binding for OpenSSL.

Once upon a time, one Mitch Kapor, who sold Lotus to IBM, and with money he got decided to perpetrate good. He did many truly good things for the computer world, was co-founder of EFF, helped with Mozilla Foundation, but he also founded rather unsuccessful project, a Python-based universal PIM, called Chandler.

Subversion of the project has been mirrored couple of times, one by me.

M2Crypto

- M2Crypto was maintained by Heikki Toivonen few years after Chandler folded, but his last release 0.21.1 was from 2011.
- Maintained in Red Hat by Miloslav Trmač.
- I took over the project in May 2015.

Maintenaned in Red Hat by Miloslav Trmač, who collected all patches in RHEL package.

I took over the project in May 2015 with the intention just to publish all patches and be a point of contact for any issue reports. I haven't expected much activity, because package was very silent in RHEL.

Strengths and weaknesses

- Backed up by stable C library
- Rather large coverage of OpenSSL API
- Surprisingly widespread use

- Large test suite
- Unknown issues
- Python 3
- M2Crypto API copies OpenSSL too closely
- Support for Mac OS X and Windows (not mentioning `*BSD`) was broken.

Backed by … comparing to PyCrypto and other horrors.

Opportunities & Threats

- Satisfying current user base
- Replacing horrors like PyCrypto
- Goal of maintenance is to maintain API
- Extend support on non-Linux platforms
- Python `ssl` module
- Python cryptography

Distribution bug tracker (especially an enterprise one) is not a good measure of the real state of use and quality of package.

There are apparently many programmers for custom software, who use M2Crypto (still it is one of the most complete bindings for OpenSSL).

Threats as a "competing" projects, which may replace M2Crypto.

Unicode

- The biggest problem of all Python 2 programs: complete confusion between py2k `str` means py3k `str` and when `bytes`.
- There are numerous uses of both in M2Crypto, because of course both strings and binary data are present in all functions of OpenSSL.

Strategy

- Type Hints
- Documentation strings
- CI
- Extension of platform support

C API

- All Unicode/bytes translation happens on C level as well
- Based on `swig`, which fortunately natively supports `--py3`.
- Also need to support two versions of OpenSSL API, 1.1 and older.
- Minimize use of `#ifdef` s and rather use included shims for missing functions.

C shims of missing functions

- For OpenSSL < 1.1
- For Python 2
  - `PyLong_FromLong()` and `PyUnicode_AsUTF8()` just simple `#define` s.
  - All Pythons >= 2.6 contain whole set of Py3k function stubs in `bytesobject.h`.
- For Python 3
  - `PyFile_AsFile()` I have no idea, why it was removed from py3k API

Type Hints

- PEP 484 providing **optional** type annotations. Quite controversial, but clearly very useful for libraries
- Native for Python >= 3.5, but supports py2k compatible syntax:

```
def sum(x, y):
    # type: (int, int) -> int
    return x + y
```

- Especially useful for our situation: marking types helps us to analyze what invidivual py2k `str` actually mean.

Python porting (shims again)

The same principles apply as with C functions, "shims, not `#ifdef` s".

- Plenty of issues are resolved by using `six` (or `modernize`, `future`), so use them. If I did it again, I would probably use `future` (and wrote py3k code), but the difference is slim.
- Do not hesitate to create your own shims. So I have for example, `bix_to_hex` and `oct_to_num` or padding functions there.
- http://python-future.org/compatible_idioms.html

Whoever does not understand LISP, is doomed to reinvent it. Badly. LISP -> six