python-sshpubkeys Documentation

Release latest

April 22, 2015

Contents

This library validates OpenSSH public keys.

Currently ssh-rsa, ssh-dss (DSA), ssh-ed25519 and ecdsa keys with NIST curves are supported.

Installation:

```
pip install sshpubkeys
```

or clone the repository and use

```
python setup.py install
```

Usage:

```
from sshpubkeys import SSHKey
ssh = SSHKey("ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAAAYQCxO38tKAJXIs9ivPxt7AY"
      "dfybgtAR1ow3Qkb9GPQ6wkFHQqcFDe6faKCxH6iDRteo4D8L8B"
      "xwzN42uZSB0nfmjkIxFTcEU3mFSXEbWByg78aoddMrAAjatyrh"
      "H1pON6P0= ojarva@ojar-laptop")
print(ssh.bits) # 768
print(ssh.hash()) # 56:84:1e:90:08:3b:60:c7:29:70:5f:5e:25:a6:3b:86
```

Exceptions

- NotImplementedError if invalid ecdsa curve or unknown key type is encountered.
- **InvalidKeyException if any other error is encountered:**
  - TooShortKeyException if key is too short (<768 bits for RSA, <1024 for DSA, <256 for ED25519)
  - TooLongKeyException if key is too long (>16384 for RSA, >1024 for DSA, >256 for ED25519)
  - InvalidTypeException if key type (“ssh-rsa” in above example) does not match to what is included in base64 encoded data.
  - MalformedDataException if decoding and extracting the data fails.

CHAPTER 2

Tests

See "tests/" folder for unit tests. Use

```
python setup.py test
```

or

```
python3 setup.py test
```

to run test suite. If you have keys that are not parsed properly, or malformed keys that raise incorrect exception, please send your *public key* to olli@jarva.fi, and I'll include it. Alternatively, create a new issue or make a pull request in github.