Samba 4 Python Scripting

Jelmer Vernooij

Samba Team

April 21, 2008
Agenda

Why Python?

Changes in the last year

Some trivial examples

Demo

Creating bindings

Future
Why scripting?

- Quicker development
- Easier to understand
  - Easy to use for system administrators
  - Lower barrier for contributions?
History of Samba and scripting

- Samba 3.0 had python bindings
  - Removed because of lack of maintainance
  - Not used for any core infrastructure
- Samba 4 has embedded JavaScript
  - Originally used for SWAT, provisioning
Why Python?

- Replaces EJS as internal scripting language
- Python is now a build-dependency
  - Easy to build from source, no dependencies
  - Ported to at least as much platforms as Samba
  - Available as standard package for most platforms
- The longer we would’ve waited, the more code we’d have to port
Why Python?

- Comes “with batteries included”
  - No need to reimplement utility functions and bindings for Samba
- Easy to create bindings
- Most existing libraries already have Python bindings
  - GTK+, Qt, HTTP, .ini-parsers...
- Large existing developer base
  - Potential contributors
- Better scripting language
  - Nested functions
  - Modularity
- More development tools available
  - Debugger, profiler, code coverage analyser, ...
What exactly is Python?

- High-level general purpose scripting language
- Object-oriented, structured or functional programming
- Memory-managed, reference-counted
- Standardsized, several implementations
- Portable
- Created in '91
What does it look like?

Hello world!

```
print "Hello World!"
```

Function usage

```
def addone(value):
    return value + 1
```

Using modules

```
import unittest
```
Some interesting software in Python

Written in Python:
- Original BitTorrent client
- Mailman
- Bazaar/Mercurial
- Trac
- Some apps in GNOME/KDE

Scriptable in Python
- Blender
- Amarok, Rhythmbox
- Vim
- Totem
- Epiphany
Python in Samba 4!

- pidl can now generate Python bindings
- SWIG used for binding several other libraries
- some bindings written manually
- now used instead of EJS in a lot of places
  - EJS still there but may be removed at a later point
Existing Samba Python bindings

- Credentials
- SamDB
- Most DCE/RPC modules
- LDB
- TDB
- Registry
- Libnet

- Includes matching unit tests, so all bindings should work
- Bindings should be Pythonic rather than one-on-one wrappers of C functions
Infrastructure in Python

- Provisioning
  - smbclient?
    - SoC student will work on this
    - Proof-of-concept will be interesting, may not be ideal as actual implementation
- SWAT
  - SoC student will work on this hopefully
- Samba-GTK
  - SoC student will work on this
Reading TDB files

```python
import tdb, sys

db = tdb.Tdb(sys.argv[1])
for (k, v) in db.items():
    print "{"" key(%d) = %r" % (len(k), k)
    print "data(%d) = %r" % (len(v), v)
    print "}""
```
Using LDB

```python
#!/usr/bin/python

import ldb

conn = ldb.Ldb("msg.tdb")

conn.add({"dn": "dc=samba,dc=org", "attr1": "foo"})

for msg in conn.search("dc=samba,dc=org"): print str(msg.dn)
```
Connecting to LDAP using LDB

```
#!/usr/bin/python

import ldb

# Connect to the LDAP server
conn = ldb.Ldb("ldap://ldap.abmas.org/")

for msg in conn.search("dc=samba,dc=org"):    
    print str(msg.dn)
```
Adding users

```python
#!/usr/bin/python
import samr, lsa

# Connect to the local SAM
conn = samr.samr("ncalrpc:", "st/dc/etc/smb.conf")

# Get SAMR connect handle
samr_handle = conn.Connect(0, 0xffffffff)

domainname = lsa.String()
domainname.string = u"SAMBADOMAIN"

sid = conn.LookupDomain(samr_handle, domainname)
print "Found sid %s for SAMBADOMAIN" % sid

conn.Close(samr_handle)
```
import winreg
from samba.tests import RpcInterfaceTestCase

class WinregTests(RpcInterfaceTestCase):
    def setUp(self):
        self.conn = winreg.winreg("ncalrpc:", self.get_loadparm(), self.get_credentials()"

    def test_hklm(self):
        handle = self.conn.OpenHKLM(None,
        winreg.KEY_QUERY_VALUE | winreg.KEYENUMERATESUBKEYS)
        self.conn.CloseKey(handle)
Demo
Demo Time
Creating bindings

- We use SWIG, http://www.swig.org/.
- Hard to grasp language but very powerful
Missing bindings

- NetBIOS
- SMB
- WINS
- CLDAP
- LDAP
Where to get?

- **ldb**
  - [http://ldb.samba.org/](http://ldb.samba.org/)
  - `python-ldb` in Debian/Ubuntu

- **tdb**
  - [http://tdb.samba.org/](http://tdb.samba.org/)
  - `python-tdb` in Debian/Ubuntu

- ...others:
  - Samba 4
  - ...not packaged yet, but hopefully soon
Learning more

- www.python.org
- `pydoc < name >`
- `pydoctor`
- Maybe public API docs on Samba.org?
Future expansions

- Port to Samba 3?
  - Need to make sure it stays maintained
- Help welcome :-(